PART II

RESEARCH PAPERS

7. MULTIRESOLUTIONAL PHENOMENA IN INTELLIGENT SYSTEMS

Heterogeneous Computing

A. Wild
Motorola, Phoenix, AZ 85018

ABSTRACT

It is often considered, explicitly or implicitly, that constructed systems with autonomy must reflect, and potentially duplicate, the perceived capabilities of human intelligence, including having the ability to handle heterogeneity, e.g. to perform numerical computing, as well as various types of non-numerical computing. Since our knowledge is getting obsolete, no matter how much wisdom will be included in an intelligent system, it will loose its sharpness in time, unless it can improve itself. This would require changing system domains and internal interfaces, and selecting architectures that would support self-change.

KEYWORDS: *Heterogeneity, non-numerical computing, domain, interface, architecture.*

1. INTRODUCTION

Several authors pointed out capabilities that an intelligent system should possess, in addition to its ability to handle numbers. According to Merriam-Webster On-line Thesaurus, “compute” has as etymology the Latin “computare”, from com- + putare, meaning “to consider”, a much wider meaning that the contemporary usage of the verb “to compute”, listed by the same source as being: 1 : to make calculation; 2 : to use a computer.

If anybody had any doubts, this historically wider perspective confirms that non-numerical computation is no oxymoron, but a legitimate area of research. However, when considering an intelligent system, it is easy to see that the question is not so much whether the right way to go is numerical or non-numerical computation : quite obviously, they should both be present among the system capabilities.

Furthermore, "non-numerical" is a simplification, reducing the number of cases to two, "tertium non datur". Actually, the generic "non-something" will spontaneously split into any number of "somethings". An intelligent system needs to be able to handle "all of the above", and more.

This paper lists some questions that have been partially addressed, from this perspective, and/or might be worth pursuing. It does not contain a corresponding list of answers, as most of them are expected from future research. It is rather initiating a whish list.

2. EVOLUTION

At any point in time, our understanding of the world is imperfect, as it is:

- incomplete, not being able to explain all we observe
- contradictory, containing different and mutually incompatible explanations of the same facts
- partially wrong, as many explanations currently accepted are likely to be falsified in the future.

We may try to incorporate all our knowledge at this point in time, or any part thereof we think appropriate, into a constructed system. But, sooner or later, unavoidably, it will become hopelessly outdated, unless it would have a built-in capability to progress. Obviously, this capability would be an important feature even at lower levels of resolution, and for simpler tasks, than maintaining an internal image of the world.

Of particular interest would be whether the system could evolve in synchronism with our understanding of the human mind, as the theories about the human mind provide a major source of inspiration for constructed systems. Their evolution would surely add new content to be implemented in constructed systems. But, in general terms, this would be the effect of any advancement, in any area of knowledge.

How can a constructed system evolve, without human intervention ? Can it include domains that became relevant after the system was built ? Can and should it modify or eliminate some of the domains implemented at its "birth", that turn out as being wrong or irrelevant ?

3. SELF-STRUCTURING

A particular aspect of the evolution is the capability to modify the interactions between the parts of the system. In an evolving system, it is to be expected that the information exchange between domains would have to be adjusted continuously, “on the fly”.

As a particular example, when evolving, the system must be able to define new interfaces between its parts, be they old or new. The changes may be initiated in response to different needs: eliminate computational bottlenecks, add new capabilities, eliminate useless parts of the system, etc. Even if no domains are added or eliminated, the system may determine that a different structure would have desirable advantages, and would take advantage of it by re-defining its own partitioning, information flow etc.

Is there a way for a system to control interfaces among its own sub-systems, e.g. define new ones, eliminate or modify existing ones ? How can a system re-architect itself ?

4. ARCHITECTURE, HIERARCHY

For some time, scientists pursued the idea that everything can be reduced to simple axioms, from which we would derive the apparent complexity of our surroundings. Still useful in particular disciplines, this hope has been largely replaced by a view accepting complexity as a fundamental feature of the world.

Usually, we handle complexity by introducing hierarchy. At each level, simpler concepts can be used to describe observations, while rules and procedures are established for crossing the boundaries from one hierarchical level to the next one.

In heterogeneous computing systems, each domain is likely to have some hierarchy, but assembling domains presents difficulties even in relatively simple cases. For instance, the hierarchical design database of an integrated circuit would not result automatically from merging hierarchical sub-circuits. If the same objects are rooted at different levels in the sub-circuits, the assembly will have most objects present at most hierarchical levels. Connections between sub-circuits will cross hierarchy borders, if the inputs/outputs of sub-circuits are at different levels. Various views of the objects in the database, such as timing, physical construction, etc., would not be propagated automatically up the hierarchy levels. To make merging possible without diluting or destroying the hierarchy, the designers of the sub-circuits must follow common, rigid rules. Alternatively, the system should automatically re-structure domain hierarchy. Today, systems do that only for the trivial case of one single level (flat hierarchy, actually no hierarchy !).

The problem is obviously more complicated when the system includes heterogeneous parts. Human programmers, exploring ad-hoc possibilities for connecting different domains of numerical computing, introduce transition domains, implementing rules for connecting space, time and parameters, e.g. by using interpolation/extrapolation rules in space, running the local times in lock step, and coding equations for parameters.

In more general terms, the requirement for an intelligent system would be to connect domains of various types of non-numerical computation, both with each other, and with domains of numerical computation.

How could a system be architected such that heterogeneous elements, like different types of computation (reasoning ?), may coexist, interact and add value to each other ? What would the interfaces between its domains be looking like ?

5. ONE, TWO, MANY

An intuitive way to build a system capable of acting upon itself is to architect it as a two-part structure: the first part addresses the tasks at hand, while the second part is optimizing the first part, acting like a conscience, or an ego of the system. This architecture seems to be able to ensure capabilities like evolving the first part of the system, or re-structuring it.

Another principle, probably much easier to envision than to implement, could build upon the paradigm of de-centralization. Any domain, facing difficulties in solving a task, would be entitled to start a browser, searching for useful capabilities in other domains. If the answer seems positive, interfaces would be put in place to connect the discovered resource with the domain trying to solve the task. If the browser does not provide a useful answer, a “generate domain” function could be started to fill the gap.

Is a non-hierarchical, self-configuring, heterogeneous system at all possible ? If yes, are there any rules to follow, are there impossible situations to avoid, or, alternatively, anything goes, and the solutions will be selected by trial and error ? Can this happen across hierarchical boundaries without generating unbearable chaos ?

6. IDENTITY, DREAMS

Allowing every domain to take the initiative in changing the system seems risky (yet democracy mostly works !). Clearly, in a two-part architecture, one part remains untouched, and can assume the task to ensure the stability of the system. In a de-centralized system, there may still be a need to define some parts as “untouchable”, and a boundary might be needed to separate them from the parts that can be changed.

For one thing, the decentralized process envisioned above is likely to accumulate, over time, numerous useless connections, unnecessary search results, and other by-products. This suggests that the system would develop a need for a cleaning procedure. The system would “go to sleep”, while running procedures that would tide it up. While “sleeping”, it would be going through a sequence of abnormal states, strange and seemingly useless, that could be metaphorically called “dreams”. The control of the system could be provided by a relatively simple and unintelligent mechanism, forcing it to undergo circadian cycles.

This line of thinking, this model and this metaphor may seem excessively anthropomorphic. Nonetheless, there may be sufficient reasons to allow for it, among other representations, in a system truly capable to handle all types of heterogeneous computing.

A Hierarchical Framework for Constructing Intelligent Systems Metrics

Ronald R. Yager
Machine Intelligence Institute
Iona College
New Rochelle, NY 10801

ABSTRACT

The focus of this work is on the development of a tool to enable the construction of performance metrics for intelligent systems which allows for the expression of intelligence in terms of high level concepts while allowing for the evaluation in terms of more basic measurable attributes.

1. Introduction

The measurement of performance of intelligent systems is clearly a context dependent process. This type of evaluation strongly depends upon the purpose for which the system is being used and the types of "intelligence" it is required to manifest.. However independent of the context the construction of such performance metrics requires the ability represent sophisticated human concepts needed to describe the various aspects of intelligence. While the expression of what constitutes intelligent performance may involve high level cognitive concepts the actual calculation of performance must be based upon measurable attributes associated with the system. The focus of this work is on the development of a tool to enable the construction of performance metrics for intelligent systems which allows for the expression of intelligence in terms of high level concepts while allowing for the evaluation in terms of more basic measurable attributes. This framework, which makes considerable use of the Ordered Weighted Averaging (OWA) operator [1, 2], also supports a hierarchical structure which allows for an increased expressiveness.

2. A General Approach to Aggregation

Central to any tool used for construction of intelligent systems metrics is the need for the aggregation of scores. In order to provide a very general framework to implement aggregations, we shall use the Ordered Weighted Averaging (OWA) operator [1, 2]. In the following, we briefly review the basic ideas associated with this class of aggregation operators.

Definition: An Ordered Weighted Averaging (OWA) operator of dimension *n* is a mapping F which has an associated weighting vector W such that its components w_j satisfy the following conditions 1. $w_j \in [0,1]$ and 2. $\sum_{j=1}^{n} w_j = 1$ and where $F(a_1, a_2,..., a_n) = \sum_{j=1}^{n} w_j b_j$ with b_j being the j^{th} largest of the a_i.

A key feature of this operator is the ordering of the arguments by value, a process that introduces a nonlinearity into the operation. Formally, we can represent this aggregation operator in vector notation as $F(a_1, a_2,..., a_n) = W^T B$, where W is the weighting vector and B is a vector, called the ordered argument vector, whose components are the b_j. Here we see the nonlinearity is restricted to the process of generating B. It can be shown that this operator is in the class of mean operators as it is commutative, monotonic, and bounded, $\text{Min}[a_i] \le F(a_1, a_2,..., a_n) \le \text{Max}[a_i]$. It can also be seen to be idempotent, $F(a, a,..., a) = a$.

The great generality of this operator lies in the fact that by selecting the w_j, we can implement many different aggregation operators. Specifically, by appropriately selecting the weights in W, we can emphasize different arguments based upon their position in the ordering. If we place most of the weights near the top of W, we can emphasize the higher scores, while placing the weights near bottom of W emphasizes the lower scores in the aggregation.

A number of special cases of these operators have been pointed out in the literature [3]. Each of these special cases is distinguished by the structure of the weighting vector W. Consider the situation where the weights are such that $w_1 = 1$ and $w_j = 0$ for all $j \neq 1$, this weighting vector is denoted as W^*. In this case we get $F(a_1, a_2,..., a_n) = \text{Max}_j[a_j]$. Thus the Max operator is a special case of the OWA operator. If the weights are such that $w_n = 1$ and $w_j = 0$ for $j \neq n$, denoted W_*, we get $F(a_1, a_2,..., a_n) = \text{Min}_j[a_j]$. Thus the Min operator is a special case of the OWA operator. If the weights are such that $w_j = \frac{1}{n}$ for all j, denoted W_{ave}, then $F(a_1, a_2,..., a_n) = \frac{1}{n}\sum_{j=1}^{n} a_j$ Thus we see that the simple average is also a special case of these operators.

If $W = W^{[k]}$ is such that $w_k = 1$ and $w_j = 0$ for $j \neq k$, then $F(a_1, a_2,..., a_n) = b_k$, the k^{th} largest of the a_i. The median is also a special case of this family of operators. If n is odd, we obtain the median by selecting $w_{\frac{n+1}{2}} = 1$ and by letting $w_j = 0$, for $j \neq \frac{n+1}{2}$. If n is even, we get the median by selecting $w_{\frac{n}{2}} = w_{\frac{n}{2}+1} = \frac{1}{2}$ and letting $w_j = 0$ for all other terms.

An interesting class of these operators is the so-called olympic aggregators. The simplest example of this case is where we select $w_1 = w_n = 0$ and let $w_j = \frac{1}{n-2}$ for $j \neq 1$ or n. In this case, we have eliminated the highest and lowest scores and we've taken the average of the rest. We note that this process is often used in obtaining aggregated scores from judges in olympic events such as gymnastics and diving.

In [1], we introduced two measures useful for characterizing OWA operators. The first of these measures, called the alpha value of the weighting vector, is defined as $\alpha = \frac{1}{n-1}\sum_{j=1}^{n} (n - j)\, w_j$. It can be shown, $\alpha \in [0, 1]$. Furthermore, it can also be shown that if $W = W^*$ then $\alpha = 1$, if $W = W_{ave}$ then $\alpha = 0.5$ and if $W = W_*$ then $\alpha = 0$.

Essentially α provides some indication of the inclination of the OWA operators for giving more weight to the higher scores or lower scores. The closer α is to one, greater preference is given to the higher scores, the closer α is to zero, the greater preference is given to lower scores, and a value close to 0.5 indicates no preference. The actual semantics associated with α depends upon the application at hand. For example, in using the OWA operators to model logical connectives between the *and* and *or*, α can be associated with a measure of the degree of *orness* associated with an aggregation. .

It can be shown that while $\alpha = 1$ only if $W = W^*$ and $\alpha = 0$ only if $W = W_*$, other values of α can be obtained for many different cases of W. A particularly interesting case is $\alpha = 0.5$. It can be shown that for any OWA operator having a W with $w_{n-j+1} = w_j$ for all j, we get $\alpha = 0.5$. Thus we see any symmetric OWA operator has $\alpha = 0.5$. Essentially these operators are in the same spirit as the simple average.

The second measure introduced in [1] was

$$Disp(W) = -\frac{1}{n-1} w_j \ln(w_j).$$

In [1] it was suggested that this measure can be used to measure the degree to which we use all the information in the argument. It can be shown that for all W

$$0 \leq Disp(w) \leq \ln(n).$$

We note $Disp(w) = 0$ iff $W = W_{(k)}$ and $Disp(w) = \ln(n)$ iff $W = W_{ave}$. It can be shown that of all the symmetric implementations of W, those having $\alpha = 0.5$ (W_{ave}) has the largest measure of Disp.

3. Linguistic Description of OWA Operators

Let us now consider a basic application of the OWA operator. Assume $A_1, A_2,...., A_n$ is a collection of measurable attributes useful in characterizing intelligence in a system. For any given system d, let $A_i(d) \in [0, 1]$ indicate the degree it satisfies the property associated with attribute A_i. Using the OWA operator we can obtain a measure of satisfaction to this collection of attributes as $Val(d) = F_w(A_i(d), A_2(d), ..., A_n(d))$. Since the value obtained as a result of using the OWA aggregation is dependent upon the weighting vector, the issue of deciding upon the weighting vector appropriate for a particular aggregation is of great importance. One of the beneficial features of the OWA operator is the considerable number of different approaches that have been suggested for obtaining the weighting vector to use in any given application [4]. Of particular significance is the strong semantic underpinning of these approaches. This strong semantic connection allows users to easily translate their requirements, which may be expressed in many different ways, into appropriate OWA weighting vectors. Here we shall describe an approach based upon the idea of linguistic quantifiers.

The concept of linguistic quantifiers was originally introduced by Zadeh [5]. A linguistic quantifier, more specifically a proportional linguistic quantifier, is a term corresponding to a proportion of objects. While most formal systems, such as logic, allow just two quantifiers, *for all* and *there exists*, as noted by Zadeh, human discourse is replete with a vast array of terms, fuzzy and crisp, that are used to express information about proportions. Examples of this are *most*, *at least half, all, about* $\frac{1}{3}$. Motivated by this Zadeh [5] suggested a method for formally representing these linguistic quantifiers. Let *Q* be a linguistic expression corresponding to a quantifier such as *most*. Zadeh suggested representing this as a fuzzy subset Q over I = [0, 1] in which for any proportion $r \in I$, Q(r) indicates the degree to which r satisfies the concept indicated by the quantifier *Q*.

In [6] Yager showed how we can use a linguistic quantifier to obtain a weighting vector W associated with an OWA aggregation. For our purposes we shall restrict ourselves to regularly increasing monotonic (RIM) quantifiers. A fuzzy subset $Q : I \rightarrow I$ is said to represent a RIM linguistic quantifier if: 1. Q(0) = 0, 2. Q(1) = 1 and 3. if $r_1 > r_2$ then $Q(r_1) \geq Q(r_2)$ (monotonic)

These RIM quantifiers model the class in which an increase in proportion results in an increase in compatibility to the linguistic expression being modeled. Examples of these types of quantifiers are *at least one, all, at least* α *%, most, more than a few, some*.

Assume Q is a RIM quantifier. Then we can associate with Q an OWA weighting vector W such that for j = 1 to n

$$w_j = Q(\frac{j}{n}) - Q(\frac{j-1}{n}).$$

Thus using this approach we obtain the weighting vector directly from the linguistic expression of the quantifier. The properties of RIMness guarantee that the properties of W are satisfied.

Let us look at the situation for some prototypical quantifiers. The quantifier *for all* is shown in figure #1. In this case we get that $w_j = 0$ for $j \neq n$, and $w_n = 1$, $W = W_*$ In this case we get as our aggregation the minimum of the aggregates. We also recall that the quantifier *for all* corresponds to the logical "anding" of all the arguments

In figure #2 we see the existential quantifier, *not none*. In this case $w_1 = 1$ and $w_j = 0$ for $j > 1$, W = W*. This can be seen as inducing the maximum aggregation. It is recalled this quantifier corresponds to a logical *oring* of the arguments

Figure #1. Linguistic quantifier "for all"

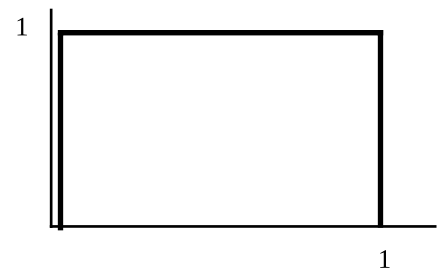


Figure #2. Linguistic quantifier "not none"

Figure #3 is seen as corresponding to the quantifier *at least* α. For this quantifier $w_j = 1$ for j such that $\frac{j-1}{n} < \alpha \leq \frac{j}{n}$ and $w_j = 0$ for all other

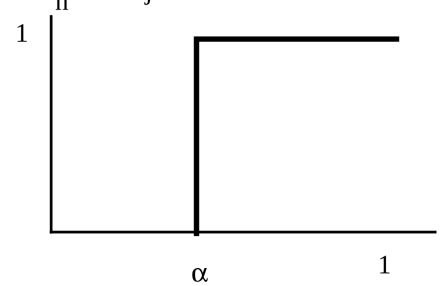


Figure #3. Linguistic quantifier "at least α"

Another quantifier is one in which Q(r) = r. Here we get $w_j = \frac{j}{n} - \frac{j-1}{n} = \frac{1}{n}$ for all j. This gives us the simple average. We denote this quantifier as *some*.

As discussed by Yager [3] one can consider parameterized families of quantifiers. Consider the parameterized family $Q(r) = r^\rho$, where $\rho \in [0, \infty]$. If $\rho = 0$, we get the existential quantifier; if $\rho = \infty$, we get *for all* and when $\rho = 1$, we are get the quantifier *some*. In addition for the case in which $\rho = 2$, $Q(r) = r^2$, we get one possible interpretation of the quantifier *most.*

As a result of the ideas so far presented here we can introduce the idea of a basic intelligence measuring module (IMM): <A_1, A_2,.... A_n: *Q*>, consisting of a collection of attributes and a linguistic quantifier indicating the proportion of the attributes we desire. Implicit in this module is the fact that the linguistic expression *Q* is essentially defining a weighting vector W for an OWA aggregation.

4. Including Attribute Importance

In the preceding we have indicated an IMM as consisting of a collection of attributes of interest and a quantifier Q indicating a mode of interaction between the attributes. Implicit in the preceding is the equal treatment of all attributes. For the construction of some intelligent systems measures we may need to ascribe differing importances to the attributes [4, 6]. In the following we shall consider the introduction of importance weights into our procedure.

Let $\alpha_i \in [0, 1]$ indicate the importance associated with attribute A_i. We assume $\alpha_i = 0$ indicates zero importance. With the introduction of these weights we can now consider a more general metric:

<A_1, A_2,...., A_n: M: Q>.

Here as before, the A_i are a collection of attributes and Q is a linguistic quantifier, however, here M is an n vector whose component $m_j = \alpha_j$, is the importance associated with A_j.

Our goal now is to calculate the overall score of a system d as $Val(d) = F_{Q/M}(A_1(d), A_2(d), ..., A_n(d))$. Here $F_{Q/M}$ indicates an OWA operator. Our agenda here will be to first find an associated OWA weighting vector, W(d), based upon both Q and M. Once having obtained this vector we calculate Val(d) by the usual OWA process) = $W(d)^T B(d) = \sum_{j=1}^{n} w_j(d)\, b_j(d)$. Here $b_j(d)$ is the j^{th} largest of the $A_i(d)$ and $w_j(d)$ is the j^{th} component of the associated OWA vector W(d)

What is important to point out here is that, as we shall subsequently see, the weighting vector will be influenced by the ordering of the $A_i(d)$.

We now describe the procedure [4, 6] that shall be used to calculate the weighting vector, $w_j(d)$. The first step is to calculate the ordered argument vector B(d) such that $b_j(d)$ is the j^{th} largest of the $A_i(d)$. Furthermore, we shall let μ_j denote the importance weight associated with the attribute that has the j^{th} largest value. Thus if $A_5(d)$ is the largest of the $A_i(d)$, then $b_1(d) = A_5(d)$ and $u_1 = \alpha_5$. Our next step is to calculate the OWA weighting vector W(d). We obtain the associated weights as $w_j(d) = Q(\frac{S_j}{T}) - Q(\frac{S_{j-1}}{T})$ where $S_j = \sum_{k=1}^{j} u_k$ and $T = S_n$. T is the sum of all the importances and S_j is the sum of the importances of the j^{th} most satisfied attributes. The following example will illustrate the use of this technique.

Example: Assume there are four attributes: A_1, A_2, A_3, A_4. The importances associated with these criteria are $u_1 = 1$, $u_2 = 0.6$, $u_3 = 0.5$ and $u_4 = 0.9$, giving us T = 3. We shall assume the quantifier guiding this aggregation is *most*, which is defined by $Q(r) = r^2$. Assume we have two system we are comparing, x and y, and the satisfactions to each of the attributes are:

$A_1(x) = 0.7$, $A_2(x) = 1$, $A_3(x) = 0.5$ and $A_4(x) = 0.6$

$A_1(y) = 0.6$, $A_2(y) = 0.3$, $A_3(y) = 0.$ and $A_4(y) = 1$

We first consider the valuation for x. In this case the ordering of the criteria satisfactions gives us:

	b_j	u_j
A_2	1	0.6
A_1	0.7	1
A_4	0.6	0.9
A_3	0.5	0.5

Calculating the weights associated with x, we get: $w_1(x) = 0.04$, $w_2(x) = 0.24$, $w_3(x) = 0.41$ and $w_4(x) = 0.31$. Using this $Val(x) = \sum_{j=1}^{4} w_j(x)\, b_j = 0.609$.

To calculate the score for y we proceed as follows. In this case the ordering of the criteria satisfaction is

	b_j	u_j
A_4	1	0.9
A_3	0.9	0.5
A_1	0.6	1
A_2	0.3	0.6

The associated weights are $w_1(y) = 0.09$, $w_2(y) = 0.13$, $w_3(y) = 0.42$and $w_4(y) = 0.36$ we then calculate

$\text{Val}(y) = \sum_{j=1}^{4} w_j(y)\, b_j = 0.567$. Using this metric we see that system x would be deemed more intelligent.

More details with respect to the properties of this methodology can be found in [4, 6], however here we shall point out some properties associated with this approach. It can be shown that any attribute that has importance weight zero no affect on the result.

Consider the situation when all the attributes have the same importance, $\alpha_j = \alpha$. In this case $w_j(d) = Q(\frac{1}{n\alpha}\sum_{k=1}^{j} \alpha) - Q(\frac{1}{n\alpha}\sum_{k=1}^{j-1} \alpha) = Q(\frac{j}{n}) - Q(\frac{j-1}{n})$. This is the same set of weights we obtained when we didn't include any information with respect to importance.

Let us now look at the form of aggregation function obtained for some special cases of linguistic quantifiers. In the following we shall assume, without loss of generality, that the indexing is such that $A_i(d) \geq A_j(d)$ if $i < j$. Furthermore we shall suppress the d and denote $A_i(d) = a_i$. Using this notational convention

$$\text{Val}(d) = F_{Q/\alpha}(a_1, a_2, \ldots a_n) = \sum_{j=1}^{n} a_j w_j$$

where $w_j = Q(\frac{1}{T}\sum_{k=1}^{j} \alpha_k) - Q(\frac{1}{T}\sum_{k=1}^{j-1} \alpha_k)$

Consider first the quantifier *some* , $Q(r) = r$. For this quantifier $w_j = \frac{\alpha_j}{T}$ and hence $\text{Val}(d) = \frac{1}{T}\sum_{j=1}^{n} \alpha_j a_j$

This is simply the weighted average of the attributes.

Consider now the case of the quantifier *for all*, $Q(1) = 1$ and $Q(r) = 0$ for $r \neq 1$. In this case $w_j = 0$ unless $\sum_{k=1}^{j} \alpha_k = T$ and $\sum_{k=1}^{j-1} \alpha_k < T$. We see $w_j = 1$ for the attribute having the smallest satisfaction and non-zero importance, hence $\text{Val}(d) = \underset{\alpha_j \neq 0}{\text{Min}} [a_j]$. For the case of the *existential* quantifier, $Q(0) = 0$ and $Q(r) = 1$ for all $r \neq 0$, we can easily show that $\text{Val}(d) = \underset{\alpha_j \neq 0}{\text{Max}} [a_j]$

Another example of a quantifier is the median quantifier. Here $Q(r) = 0$ for $r < 0.5$ and $Q(r) = 1$ for $r \geq 0.5$. In this case it can be shown that Val(d) can be obtained by the following simple process. First we normalize the weights, $\hat{\alpha}_j = \frac{\alpha_j}{T}$. Next we order the attribute scores in descending order and associate with each its normalized weight. We then, starting from the top, the highest score, add the normalized weights until we first reach a total of 0.5, the score of that attribute at which this total is reached is the aggregated value.

An interesting example of an OWA aggregation is the olympic aggregation. Here $w_1 = w_n = 0$ and $w_j = \frac{1}{n-2}$ for $j \neq 1$ or n. Using this aggregation we eliminate the highest and lowest scores and then take the average of the remaining scores. We can provide a generalization of this type of aggregation using a quantifier shown in figure #4.

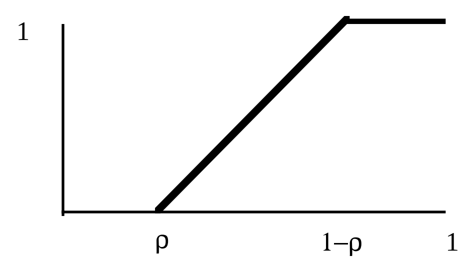


Figure #4. Generalized Olympic Quantifier

For this case

$$Q(r) = 0 \quad r < \rho$$
$$Q(r) = \frac{r - \rho}{1 - 2\rho} \qquad \rho \leq r \leq 1 - \rho$$
$$Q(r) = 1 \quad r > 1 - \rho$$

Here $w_j = 0$ for all j for which $\sum_{k=1}^{j} \frac{\alpha_k}{T} < \rho$ Similarly, $w_j = 0$ for all j for which $\sum_{k=j}^{n} \frac{\alpha_k}{T} > 1 - \rho$. In the range in between $w_j = \frac{\alpha_j}{1 - 2\rho}$

Another interesting example of OWA aggregation, one that is in some sense a dual of the olympic aggregation, is the so called Arrow-Hurwicz aggregation [7]. Here $w_1 = \alpha$ and $w_n = 1 - \alpha$, and $w_j = 0$ for all other. In this case we just consider the extreme values and eliminate the middle values. We can provide a generalization of this type of aggregation, one that can be used with importance weighted attributes, by introducing the quantifier shown in figure#5.

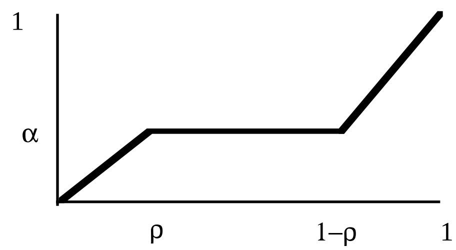


Figure #5. Generalized Arrow-Hurwicz

For this quantifier: $Q(r) = \frac{\alpha}{\rho}\ r$ if $r < \rho$, $Q(r) = \alpha$ if $\rho \leq r < 1 - \rho$ and $Q(r) = 1 - \frac{1 - \alpha}{\rho}(1 - r)$ if $r \geq 1 - \rho$. It is assumed $\rho \leq 0.5$. For this quantifier the weights used in the OWA aggregation are such that for the highest scoring attributes, those accounting for ρ portion of the importance, $w_j = \frac{\alpha}{\rho}$, for the least satisfied attributes, those accounting for ρ portion of the importance, $w_j = \frac{1 - \alpha}{\rho}$ and the middle scoring attributes $w_j = 0$. In this quantifier α can be seen as a degree of optimism and 1 - ρ as an indication of the extremism of the aggregation. A number special cases of this quantifier are worth noting. If $\rho = 0$ then we have $w_1 = \alpha$ and $w_n = 1 - \alpha$, the basic Arrow-Hurwicz aggregation. If $\alpha = \rho = 0.5$ then we get the quantifier $Q(r) = r$. If $\alpha = 1$ then we get the quantifier *at least ρ* and if $\alpha = 0$ then we get the quantifier *at least 1 – ρ.*

5. Including Priorities

In the preceding we have described a method for measuring the intelligence of a system based upon the metric $<A_1, A_2, ..., A_n: M: Q>$where the component α_j of the vector M indicates the weight associated with the attribute A_j. Implicit in our formulation was the idea that the weight α_j was explicitly provided by the user. This is not necessarily required. It is possible for the weight associated with attribute A_j to be determined by some property of the system itself. Thus let B_j be some measurable attribute associated with a system, and let $B_j(d)$ be the degree to which d satisfies this attribute. Then without introducing any additional complexity we can allow $\alpha_j(d) = B_j(d)$. Thus here the weight associated with attribute A_j depends the value $B_j(d)$. Thus within this framework we have the option of specifying the importance weights conditionally or non-conditionally or not at all.

Typically the association of importance weights with attributes reflects some measure of trade-off between the worth of the attributes. For example, consider the averaging operator where $Val(d) = \sum_{j=1}^{n} A_j(d)\ \alpha_j$. We see that a gain of Δ in $A_j(d)$ results in an increase in overall evaluation of $\alpha_j\ \Delta$, while a gain of Δ in $A_i(d)$ is worth an increase of $\alpha_i\ \Delta$. In particular, if $\alpha_j = 2$ and $\alpha_i = 1$, then we are willing to trade a gain of Δ in A_j for a loss of less than 2Δ in A_i.

In some cases where we desire two attributes, we may not be willing to trade-off one of for the other. For example, in evaluating the performance of an "intelligent" car, while we would like both safety and mileage efficiency, we are not willing to give up safety for efficiency. Such a situation implies the existence of a **priority** between the attributes, safety has priority over cost. In the following we suggest a mechanism for the inclusion of priority type relationships.

Assume A_1 and A_2 are two attributes for which there exists a priority relationship: A_1 has priority over A_2. In order to manifest this priority relationship, we require that the importance associated with A_2 be dependent upon the satisfaction of attribute A_1 by the system being evaluated. Here then $\alpha_2(d) = A_1(d)$. Let us investigate this idea for the simple weighted average. Assuming α_1 is fixed, we get

$$Val(d) = \alpha_1\ A_1(d) + A_1(d)\ A_2(d) = A_1(d)(\alpha_1 + A_2(d))$$

Here we see that if $A_1(d)$ is low, the contribution of $A_2(d)$ becomes small and hence it is not possible for a high value of A_2 to compensate for a low satisfaction to A_1. Thus if a system scores low on A_1 it will get a low rating.

More generally, consider the quantifier Q and assume A_i has priority over A_j. To implement this priority we make the importance associated with A_j related to the satisfaction of A_i. In particular, we let $\alpha_j = \alpha\ A_i$, where $\alpha \in [0, 1]$. Using this we get for the weight w_j associated with A_j that

$$w_j = Q(S_{k-1} + \frac{\alpha\ A_i(d)}{T}) - Q(S_{k-1})$$

where $S_{k-1} = \frac{\sum_{k=1}^{j-1} \alpha_k}{T}$. We see that as $A_i(d)$ gets smaller, the value w_j will decrease.

6. Concepts and Hierarchies

Throughout the preceding we have assumed a collection of attributes characterized by the fact that for any d we have available $A_i(d) \in [0,1]$, we say that the value of attribute A_j is directly accessible, we call A_j ground attribute. We shall now introduce a more general idea which we shall call an **I**ntelligence **M**easuring ***Concept*** **(*IM Concept*)**. We define an IM Concept as an object whose measure of satisfaction can be obtained for any system d. It is clear that the ground attributes are examples of concepts, they are special concepts in that their values are directly accessible.

Consider now the intelligence measuring module of the type we have previously introduced, it is of the form $<A_1, A_2, ..., A_q: M: Q>$. As we have indicated the evaluation of this for any system d can be obtained using our aggregation process. In the light of this observation, we can consider this object to be a IM concept, with Con = $<A_1, A_2, ..., A_q: M: Q>$ then its evaluation is Con(d) = $F_{Q/M}(A_1(d), A_2(d), ..., A_q(d))$. A special concept is an individual attribute, Con = $<A_j : M: Q> = A_j$,we shall call these atomic concepts. These atomic concepts require no Q or M, but just need an A_j specification.

The basic componentsat these IM Concepts are the attributes, the A_j. However, from a formal point of view, the ability to evaluate these type of concepts is based upon the fact that for any d we have a value $A_j(d)$. As we have just indicated a concept also has this property, for any d we can obtain a measure of its satisfaction. This observation allows us to extend our idea of IM concepts to allow for IM concepts whose evaluation depends upon other concepts. Thus we can consider IM concepts of the form

Con = $<Con_1, Con_2, ..., Con_n: M: Q>$.

Here each of the Con_j are concepts used to determine the satisfaction of Con by an aggregation process where M determines the weight of each of the participating concepts and Q is the quantifier guiding the aggregation of the component concepts.

The introduction of concepts into the intelligence measuring results in a hierarchical structure for the construction of metrics. Essentially, we unfold until we end up with atomic attributes which we can directly evaluate. The following simple examples illustrate the structure.

Example: Consider here the measure

$(A_1$ and A_2 and $A_3)$ or $(A_3$ and $A_4)$.

We consider this as a concept $<Con_1, Con_2 : M: Q>$.

Here Q is the existential quantifier and $M = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$ In addition

$Con_1 = <A_1, A_2, A_3: M_1: Q_1>$

$Con_2 = <A_3, A_4 : M_2: Q_2>$

Here $Q_1 = Q_2 =$ *all* and $M_1 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$ and $M_2 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$.

This can be expressed in a hierarchical fashion, see figure #6.

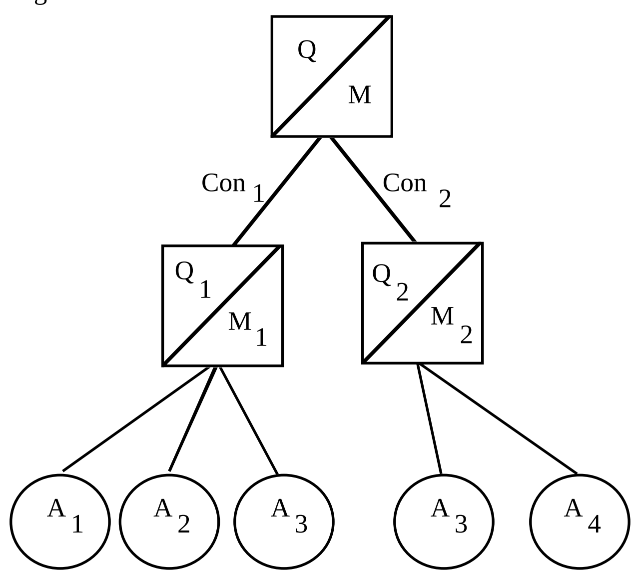


Figure #6. Hierarchical Formulation

An often used construct is the logical **if ... then** specification expressing the desire for some attribute if some other attribute is present. In a the following we describe a method for modeling this type of structure within our our framework.

Consider (A_1 **and** A_2) **or** (**if** A_3 **then** A_4). Figure #7 provides its hierarchical expansion.

In constructing this hierarchical implementation, we used the fact that “if A_3 then A_4” is logically equivalent to “not(A_3) or A_4.” Thus in this framework we interpret the concept "**if** A **then** B" as the concept $\bar{A}$ **or** B. We note that $\bar{A}(d) = 1 - A(d)$. More generally, the expression

if A_1 **and** A_2 **and** A_3 **then** B

is seen as equivalent to the expression $\bar{A}_1$ **or** $\bar{A}_2$ **or** $\bar{A}_3$ **or** B. This is represented as concept of the form <$\bar{A}_1$ $\bar{A}_2$, $\bar{A}_3$, B: -:*Or*>. We note the importances have not been specified and hence by default are all assumed to be one.

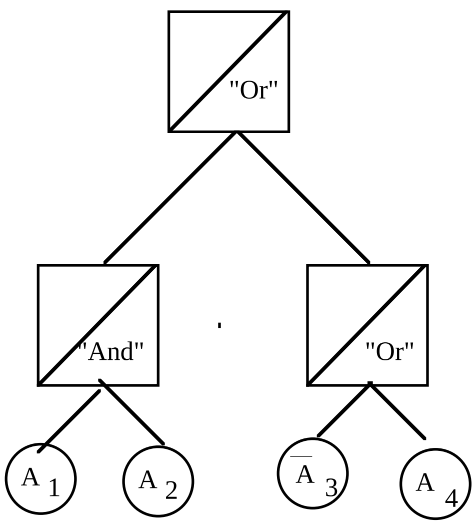


Figure #7. Implementation of if ... then

Using the ideas presented in the preceding we have a tool that can be used to construct complex measures of intelligent system performance. Using this tool we start with a high level expression of the appropriate measure. We then decompose this expression into the aggregation simpler concepts. We proceed in this manner until we obtain a characterization of our desired measure in terms of ground attributes which can be directly measured.

7. References

[1]. Yager, R. R., "On ordered weighted averaging aggregation operators in multi-criteria decision making," IEEE Transactions on Systems, Man and Cybernetics 18, 183-190, 1988.

[2]. Yager, R. R. and Kacprzyk, J., The Ordered Weighted Averaging Operators: Theory and Applications, Kluwer: Norwell, MA, 1997.

[3]. Yager, R. R., "Families of OWA operators," Fuzzy Sets and Systems 59, 125-148, 1993.

[4]. Yager, R. R., "On the inclusion of importances in OWA aggregations," in The Ordered Weighted Averaging Operators: Theory and Applications, edited by Yager, R. R. and Kacprzyk, J., Kluwer Academic Publishers: Norwell, MA, 41-59, 1997.

[5]. Zadeh, L. A., "A computational approach to fuzzy quantifiers in natural languages," Computing and Mathematics with Applications 9, 149-184, 1983.

[6]. Yager, R. R., "Quantifier guided aggregation using OWA operators," International Journal of Intelligent Systems 11, 49-73, 1996.

[7]. Arrow, K. J. and Hurwicz, L., "An optimality criterion for decision making under ignorance," in Uncertainty and Expectations in Economics, edited by Carter, C. F. and Ford, J. L., Kelley: New Jersey, 1972.

EXPLORATORY ANALYSIS ENABLED BY MULTIRESOLULTION, MULTIPERSPECTIVE MODELING

Paul K. Davis

RAND and the RAND Graduate School
1700 Main St., Santa Monica, CA 90407-2138, U.S.A.

ABSTRACT

The objective of exploratory analysis is to gain a broad understanding of a problem domain before going into details for particular cases. Its focus is understanding comprehensively the consequences of uncertainty, which requires a good deal more than normal sensitivity analysis. Such analysis is facilitated by multiresolution, multiperspective modeling (MRMPM) structures that are becoming increasingly practical. A knowledge of related design principles can help build interfaces to more normal legacy models, which can also be used for exploration.

1 BACKGROUND

Strategy problems are typically characterized by enormous uncertainties that should be central in assessment of alternative courses of action—although individuals and organizations often suppress those uncertainties and give a bizarre level of credence to wishful-thinking planning factors and other simplifications (Davis, 1994; Ch. 4; Davis, Gompert, and Kugler, 1996). In the past, an excuse for downplaying uncertainty analysis—except for marginal sensitivity analysis around some "best-estimate" baseline of dubious validity—was the sheer difficulty of doing better. The time required for setup, run, and analysis made extensive uncertainty work infeasible. Today, technology permits extensive uncertainty analysis with personal computers.

A key to treating uncertainty well is *exploratory analysis* (Davis and Hillestad, 2001). The objectives of exploratory analysis include understanding the implications of uncertainty for the problem at hand and informing the choice of strategy and subsequent modifications. In particular, *exploratory analysis can help identify strategies that are flexible, adaptive, and robust.* In successive sections, this paper describes exploratory analysis; puts it in context; discusses enabling technology and theory; points to companion papers applying the ideas; and concludes with some technology challenges for modeling and simulation. The paper draws heavily on a forthcoming book (Davis and Hillestad, 2001) and builds on a much rougher preliminary presentation of the same material (Davis, 2000).

2 EXPLORATORY ANALYSIS

2.1 What Exploratory Analysis Is and Is Not

Exploratory analysis examines the consequences of uncertainty. It can be thought of as sensitivity analysis done right, but is so different from usual sensitivity analysis as to deserve a separate name. It is closely related to scenario space analysis (Davis, 1994, Ch. 4) and "exploratory modeling" (Bankes, 1993; Lempert, et al., 1996). It is particularly useful for gaining a broad understanding of a problem domain before dipping into details. That, in turn, can greatly assist in the development and choice of strategies. It can also enhance "capabilities-based planning" by clarifying *when*—i.e., in what circumstances and with what assumptions about all the other factors—a given capability such as an improved weapon system or enhanced command and control will likely be sufficient or effective (Davis, Gompert, and Kugler, 1996). This contrasts with establishing a base-case scenario, and an organizationally blessed model and data base, and then asking "How does the outcome change if I have more of this capability?"

2.2 Types of Uncertainty

Uncertainty comes in many forms and it is useful (National Research Council, 1997) to distinguish between input uncertainties (i.e., parametric uncertainties) and structural uncertainty. Input uncertainty relates to imprecise knowledge of the model's input values. Structural uncertainty relates to questions about the form of the model itself: Does it reflect all the variables on which the real-world phenomenon purportedly described by the model depends? Is the analytical form correct? Some uncertainties may be inherent because they represent stochastic processes. Some may relate to fuzziness or imprecision, while others reflect discord among experts. Some relate to knowledge about the values of well-defined parameters, whereas others refer to future values that as yet have no true values.

It is convenient to express the uncertainties parametrically. If unsure about the model's form, we can describe this also to some extent with parameters. For example, parameters may control the relative size of quadratic and exponential terms in an otherwise linear model. Or a discrete parameter may be a switch choosing among distinct analytical forms. Some parameters may apply to the deterministic aspect of a model, others to a stochastic aspect. For example, a model might describe the rate at which Red and Blue suffer attrition in combat according to a simplistic Lanchester square law:

$$\frac{d\tilde{R}}{dt} = -\tilde{K}_b \tilde{B}(t) \quad \frac{d\tilde{B}}{dt} = -\tilde{K}_r \tilde{R}(t)$$

where the attrition coefficients for Red and Blue have both deterministic and stochastic parts, each of which are subject to uncertainty, as in (illustrating for Blue only)

$$\tilde{K}_b(t) = K_{bo}[1 + c_b \tilde{N}_b(t; \mu, \sigma_b)].$$

Here the N term is a normal random variable with mean μ and standard deviation σ. It represents stochastic processes occurring within a particular simulated war, e.g., from one time period to the next. The means and standard deviations are ordinary deterministic parameters, as are the coefficients K_{bo}, K_{ro}, c_r, and c_b. These have constant values within a particular war, but at what value they are constant is uncertain.

So far the equations have represented input uncertainty. However, suppose there is controversy over using the linear, square, or some hybrid version of a Lanchester equation. We could represent this dispute as input, or parametric, uncertainty by modifying the equation to read

$$\frac{d\tilde{R}}{dt} = -\tilde{K}_b \tilde{B}^e(t)\tilde{R}^f(t) \quad \frac{d\tilde{B}}{dt} = -\tilde{K}_r \tilde{B}^g(t)\tilde{R}^h(t).$$

Now, by treating the exponents as uncertain parameters, we could explore both input and structural uncertainties in the model—at least to some extent. The fly in the ointment is that nature's combat equations are much more complex (if they exist), and we don't even know their form. Suppose, merely as an example, that combatant effectiveness decays exponentially as combatants grow weary. We could not explore the consequences of different decay times if we did not even recognize the phenomenon in the equation's form. In fact, we *often* do not know the true system model. Nonetheless, much can be accomplished by allowing for diverse effects parametrically.

2.3 Types of Exploratory Analysis

Exploratory analysis can be conducted in several ways (Davis and Hillestad, 2001). Although most of the methods have been used in the past (see especially Morgan and Henrion, 1992), they are still not appreciated and are often poorly understood.

Input exploration (or *parametric exploration*) involves conducting model runs across the space of cases defined by discrete values of the parameters within their plausible domains. It considers not just excursions taken one-at-a-time as in normal sensitivity analysis relative to some presumed base-case set of values, but rather all the cases corresponding to value combinations defined by an experimental design (or a smaller sample). The results of such runs, which may number from dozens to hundreds of thousands or more, can be explored interactively with modern displays. Within perhaps a half-hour, a good analyst doing such exploration can often gain numerous important insights that were previously buried. He can understand not just which variables "matter," but *when*. For example, he may find that the outcome of the analysis may be rather insensitive to a given parameter for the so-called base case of assumptions, but quite sensitive for other plausible assumptions. That is, he may identify in what cases the parameter is important. To do capabilities-based planning for complex systems, this can be distinctly nontrivial.

A complement to parametric exploration is "*probabilistic exploration"* in which uncertainty about the input parameters is represented by distribution functions representing the totality of one's so-called objective and subjective knowledge. I sometimes use quotes around "probability" because the distributions are seldom true frequencies or rigorous Bayesian probabilities, but rather rough estimates or analytical conveniences.

Using analytical or Monte Carlo methods, the resulting distribution of outcomes can be calculated. This can quickly give a sense for whether uncertainty is particularly important. In contrast to displays of parametric exploration, the output of probabilistic exploration gives little visual weight to improbable cases in which various inputs all have unlikely values simultaneously. Probabilistic exploration can be very useful for a condensed net assessment. Note that this use of probability methods is different from using them to describe the consequences of a stochastic process within a given simulation run. Indeed, one should be cautious about using probabilistic exploration because one can readily confuse variation across an ensemble of possible cases (e.g., different runs of a war simulation) with variation within a single case (e.g., fluctuation from day to day within a single simulated war). Also, an unknown constant parameter for a given simulated war is no longer unknown once the simulation begins and simulation agents representing commanders should perhaps observe and act upon the correct values within a few simulated time periods. Despite these subtleties, probabilistic exploration can be quite helpful.

The preferred approach treats some uncertainties parametrically and others with uncertainty distributions. That is, it is *hybrid exploration.* It may be appropriate to parameterize a few key variables that are under one's own control (purchases, allocation of resources, and so on), while treating the uncertainty of other variables through uncertainty distributions. One may also want also to parameterize a few variables characterizing the future context in which strategy must operate (e.g., short warning time). There is no general procedure here; instead, the procedure should be tailored to the problem at hand. In any case, the result can be a comprehensible summary of how known classes of uncertainty affect the problem at hand.

Let me give a few examples of what exploratory analysis can look like. Figure 1 mimics a computer screen during a parametric exploration of what is required militarily to defend Kuwait against a future Iraqi invasion by interdicting the attacker's movement with aircraft and missiles (Davis and Carrillo, 1997). Each square denotes the outcome of a particular model case (i.e., a specific choice of all the input values). The model being used depends on 10 variables–those on the x, y, and z axes, and seven listed to the side (the z–axis variable is also listed there, redundantly). The outcome of a given simulation is represented by the color (or, in this paper, by the pattern) of a given square. Thus, a white square represents a good case in which the attacker penetrates only a few tens of kilometers before being halted. A black square represents a bad case in which the attacker penetrates deep into the region that contains critical oil facilities. The other patterns represent in-between cases. The number in each square gives the penetration distance in km.

To display results in this way for a sizable scenario space RAND has often used a program called Data View, developed at RAND in the mid 1990s by Stephen Bankes and James Gillogly. After running the thousands or hundreds of thousands of cases corresponding to an experimental design for parametric exploration, we explore the outcome space at the computer. We can choose interactively which of the parameters to vary along the x, y, and z axes of the display. The other parameters then have the values shown along the right. However, we can click on their values and change them interactively by selecting from the menu of values for cases that have been run.

As mentioned above, in about a half an hour of such interactive work, one can develop a strong sense of how outcomes vary with *combinations* of parameter values. This is much more than traditional sensitivity analysis. Moreover, one can search out and focus upon the "good" cases. Figure 1 is merely one schematic snapshot of the computer screen for choices of parameter values that show some successes. Most snapshots would be dominated by black squares because it is difficult to defend Kuwait against a large threat. Data View is not a commercial product, but RAND has made it available to government clients and some other organizations (e.g., allied military staffs).

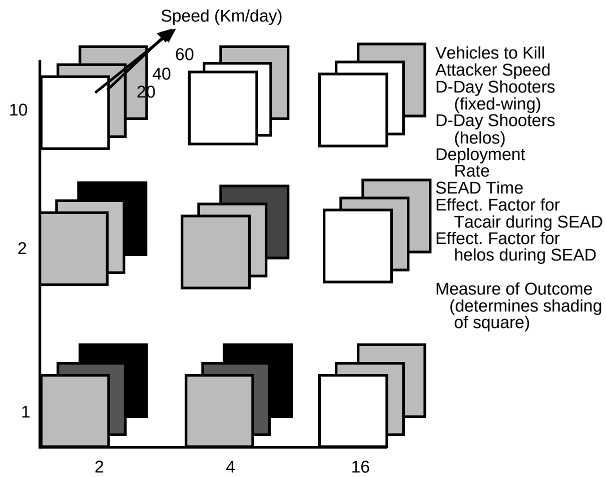


Figure 1: Display of Parametric Exploration

Other personal-computer tools can be used for the same purpose and the state of the art for such work is advancing rapidly. A much improved version of Data View called CAR™ is under development by Steve Bankes at Evolving Logic (`www.evolvinglogic.com`). For those who prefer spreadsheet modeling, there are plug-in programs for Microsoft EXCEL® that provide statistical capabilities and some means for exploratory analysis. Two of them are Crystal Ball® (`www.decisioneering.com`) and @Risk® (`www.palisade.com/html/risk.html`). For a number of reasons such as visual modeling and convenient array mathematics, I usually prefer the Analytica® modeling system (the exception is when one needs procedural programming). Analytica (`www.lumina.com`) is an outgrowth of the Demos system developed at Carnegie Mellon University (Morgan and Henrion, 1992).

Figure 2 shows a screen image from recent work with Analytica on the same problem treated in Figure 1. In this case, we have a more traditional graphical display. Outcome is measured along the Y axis and one of the independent variables is plotted along the X axis. A second variable (D-Day shooters) is reflected in the family of curves. The other independent variables appear in the rotation boxes at the top. As with Data View, we change parameter values by clicking on a value and selecting from a menu of values. Such interactive displays allow us to "fly through the outcome space" for many independent parameters, in this case 9. For this number, the display was still quickly interactive for the given model and computer (a Macintosh PowerBook G3 with 256 MB of RAM).

So far, the examples have focused on parametric exploration. Figure 3 illustrates a hybrid exploration (Davis, et al., 1998). It shows the distribution of simulation outcomes resulting from having varied most parameter values "probabilistically" across an ensemble of possible wars, but with warning time and the delay in attacking armored columns left parametric.

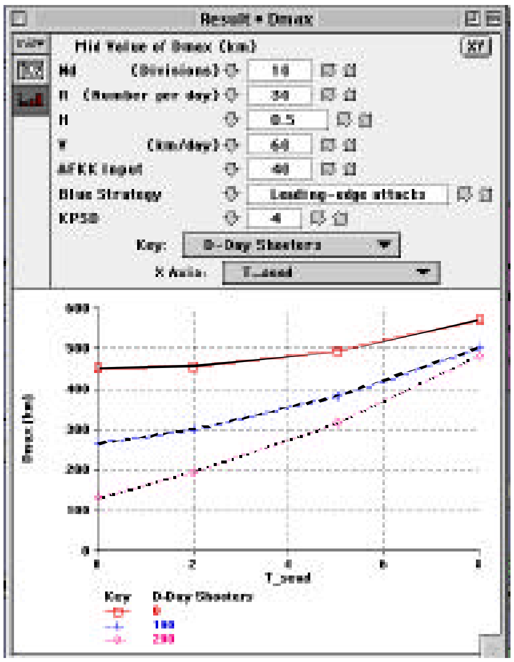


Figure 2: Analytica Display of Parametric Exploration

The probabilistic aspect of the calculation assumed, for example, that the enemy's movement rate had a triangular distribution across a particular range of values and that the suppression of air defenses would either be in the range of a few days or more like a week, depending on whether the enemy did or did not have air-defense systems and tactics that were not part of the best estimate. We represented this possibility with a discrete distribution for the likelihood of such surprises. The two curves in Figure 3 differ in that the one with crosses for markers assumes that interdiction of moving columns waits for suppression of air defenses (SEAD). The other curve assumes that interdiction begins immediately because the aircraft are assumed stealthy.

This depiction of the problem shows how widely the outcomes can vary and how the outcome distribution can be complex. The non-stealthy-aircraft case shows a spike at the right end where cases pile up because, in the simulation, the attacker halts at an objective of about 600km. Note that the mean is not a good metric: the "variance" is huge and the outcome may be multimodal.

These results have been from analyses accomplished in recent years for the Department of Defense. As we look to the future, much more is possible with computational tools. Much better displays are possible for the same information and, even more exciting, computational tools can be used to aid in the search process of exploration. For example, instead of clicking through the regions of the outcome space, tools could automatically find portions of the space in which particular outcomes are found. One could then fine-tune one's insights by clicking around in that much more limited region of the outcome space. Or, if the model is itself driven by the exploration apparatus, then the apparatus could search for outcomes of interest and then focus exploration on those regions of the input space. That is, the experimental design could be an output of the search rather than an input of the analysis process. These methods are at the core of the evolving tool mentioned earlier called CAR (for Computer-Assisted Reasoning).

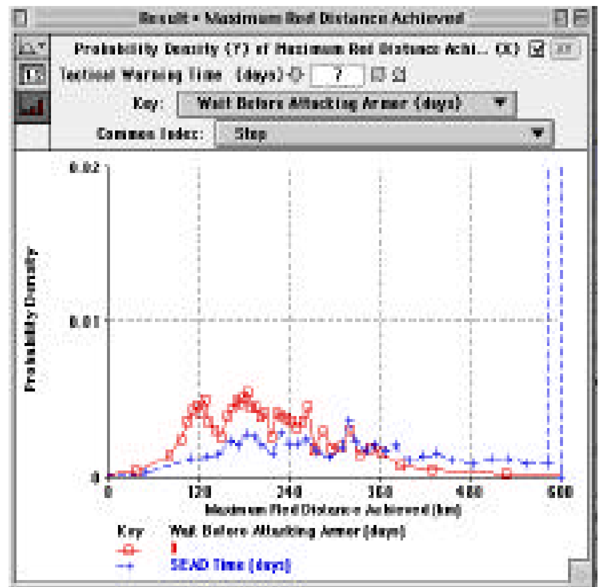


Figure 3: Analytica Display of "Probabilistic" Exploration

3 EXPLORATORY ANALYSIS IN CONTEXT

Exploratory analysis is an exciting development with a long history with RAND's RSAS and JICM models. However, it is only one part of a sound approach to analysis generally. It is worth pausing to emphasize this point. Figure 4 shows how different types of models and simulations (including human games) have distinct virtues. The figure is specialized to military applications, but a more generic version applies broadly to a wide class of analysis problems.

White rectangles indicate "good;" that is, if a cell of the matrix is white, then the type model indicated in the left column is very effective with respect to the attribute indicated in the cell's column. In particular, analytical models (top left corner), which have low resolution, can be especially powerful with respect to their analytical agility and breadth. In contrast, they are very poor (black cells) with respect to recognizing or dealing with the richness of underlying phenomena, or with the consequences of both human decisions and behavior. In contrast, field experiments often have very high resolution (they may be using the real equipment and people), and may be good or very good for revealing phenomena and reflecting human issues. They are, however, unwieldy and inappropriate for studying issues in breadth. The small insets in some of the cells indicate that the value of the type model for the particular purpose can often be enhanced a notch or two if the models include sensible decision algorithms or knowledge-based models that might be in the form of expert systems or artificial-intelligence agents.

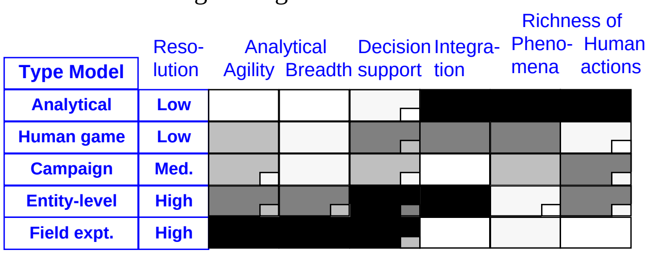


Figure 4: Virtues of a Model and Gaming Family

Figure 4 was developed as part of an exhortation to the Department of Defense regarding the need to have *families of models* and *families of analysis* (Davis, Bigelow, and McEver, 1999). Unfortunately, government agencies often focus on a single model such as the venerable TACWAR, BRAWLER, or JANUS.

The niche of exploratory analysis is the top left hand corner of the matrix in Figure 4, which emphasizes analytical agility and *breadth* of analysis, rather than depth. However, the technique can be used hierarchically if one has a suitably modularized system model. One can do top-level exploration first and then zoom in. This is easier said than done, however, especially with traditional models. Specially designed models make things much easier, as discussed in what follows.

4 TECHNOLOGICAL ENABLERS

4.1 The Curse of Dimensionality

In principle, exploratory analysis can be accomplished with any model. In practice, it becomes difficult with large models. If F represents the model, it can be considered to be simply a complicated function of many variables. How can we run a computerized version of F to understand its character? If F has M inputs with uncertain values, then we could consider N values for each input, construct a full factorial design (or some subset, using an experimental design and sampling), run the cases, and thereby have a characterization. However, the number of such cases would grow rapidly (as N^M for full-factorial analysis), which quickly gets out of hand even with big computers. Quite aside from setup-and-run-time issues, comprehending and communicating the consequences becomes very difficult if M is large. Suppose someone asked “Under what conditions is F less than the danger point?” Given sufficiently powerful computers and enough time, we could create a data base of all the cases, after which we could respond to the question by spewing out lists of the cases in which F fell below the danger point. The list, however, might go on for thousands of pages. What would we do with the list? This is one manifestation of the curse of dimensionality.

4.2 The Need for Abstractions

It follows that, even if we have a perfect high-resolution model, we need abstractions to use it well. And, in the dominant case in which the high-resolution model is by no means perfect, we need abstractions that allow us to ponder the phenomena in meaningful ways, with relatively small numbers of cognitive chunks. People can reason with 3, 5, or 10 such cognitive chunks at a time, but not with hundreds. If the problem is truly complex, we must find ways to organize our reasoning. That is, we must decompose the problem by using principles of modularity and hierarchy. The need for an aspect of hierarchical organization is inescapable in most systems of interest—even though the system may be highly distributed and relatively nonhierarchical in an organizational sense.

A corollary of our need for abstractions is that *we need models that use the various abstractions as inputs*. It is not sufficient merely to display the abstracts as intermediate outputs (displays) of the ultimate detailed model. The reasons include the fact that when a decision maker asks a what-if question using abstractions, there is a 1:n mapping problem in translating his question into the inputs of a more detailed model. So also when one obtains macroscopic empirical information and tries to use it for calibration. Although analysts can trick the model by selecting a mapping, doing so can be cumbersome and treacherous. It is often better if the question can be answered by a model that accepts the abstractions as inputs.

4.3 Finding the Abstractions

Given the need for abstractions, how do we find them and how do we exploit them? Some guidelines are emerging (Davis and Bigelow, 1998).

4.3.1 When Conceiving New Models and Families

With new models, the issue is how to *design*. Several options here are as follows:

- Design the models and model families top down so that significant abstractions are built in from the start, but do so with enough understanding of the microscopics so that the top-down design is valid.

- Design the models and families bottom up, but with enough top-down insight to assure good intermediate-level abstractions from the start.
- Do either or both of the above, but with designs taken from different perspectives.

The list does not include a pure top-down or pure bottom-up design approach. Only seldom will either generate a good design of a complex system. Note also the idea of alternative perspectives. For example, those in combat arms may conceive military problems differently than logisticians, and even more differently than historians attempting a macro-view explanation of events.

4.3.2 When Dealing With Existing Models

Only sometimes do we have the opportunity to design from scratch. More typically, we must adapt existing models. Moreover, the model "families" we may have to work with are often families more on the basis of assertion than lineage. What do we then do? Some possibilities here are:

- Study the model and the questions that users ask of the model to discover useful abstractions. For example, inputs X, Y, and Z may enter the computations only as the product XYZ. Or a decision maker may ask questions in terms of concepts like force ratio. For mature models, the displays that have been added over time provide insights into useful abstractions.
- Apply statistical machinery to search for useful abstractions. For example, such machinery might test to see whether the system's behavior correlates not just with X,Y, and Z, but with XY, XZ, YZ, or XYZ.
- Idealize the system mathematically and combine this with physical insight or empirical observation to guess at the form of aggregate behavior (e.g., inverse dependence on one variable, or exponential dependence on another). Consider approximations such as an integral being the product of the effective width of the integration interval and a representative non-zero value of the integrand.

The first approach is perhaps a natural activity for a smart modeler and programmer who begins to study an existing program, but only if he open-minded about the usefulness of higher-level depictions. The second approach is an extension of normal statistical analysis. The third approach is a hybrid that I typically prefer to the second. It uses one's understanding of phenomenology, and theories of system behavior, to gain insights about the likely or possible abstractions *before* cranking statistical machinery.

4.3.3 The Problem with Occam's Razor

The principle of Occam's razor requires that we prefer the simplest explanation and, thus, the simplest model. Enthusiasts of statistical approaches tend to interpret this to mean that one should minimize the number of variables. They tend to focus on data and to avoid adding variables for "explanation" if the variables are not needed to predict the data. In contrast, subject-area phenomenologists may prefer to enrich the depiction by adding variables that provide a better picture of cause-effect chains, but go well beyond what can be supported with meager experimental data. My own predilection is that of the phenomenologist, but with MRM designs one can sometimes have one's cake and eat it: one can test results empirically by focusing on the abstract versions of a model, while using richer versions for deeper explanation.

As an aside, a version of the Occam's Razor principle emphasizes use of the explanation that is simplest enough to explain all there is to explain, but nothing simpler! This should include phenomena that one "knows about" even if they are not clearly visible in the limited data. I would add to this the admonition made decades ago by MIT's Jay Forrester that to omit showing a variable explicitly may be equivalent to assuming its value is unity.

Competition among approaches can be useful. For example, phenomenologists working a problem may be convinced that a problem must be described with complex computer programs having hundreds or thousands of data elements. A statistical analysis may show that, despite the model's apparent richness, the system's resulting behavior is driven by something much simpler. This, in turn, may lead to a reconceptualizing of the problem phenomenologically. Many analogues exist in physics and engineering.

4.3.4 Connections Between New and Old Models

Although the discussion in Section 4.3.2 distinguished sharply between the case of new models and old ones, the reader may have noticed connections. In essence, working with existing models should often involve sketching what the models *should* be like and how models with different resolution *should* connect substantively. That is, working with existing models may require us to go back to design issues. Individuals differ, but I, at least, often find it easier to engage the problem than to engage someone's else's idiosyncratically described solution. Furthermore, I then have a better understanding of assumptions and approximations.

With this background, let me now turn to the design of multiresolution, multiperspective models and families (Davis and Bigelow, 1999). Although this relates most directly to new models, it is relevant also to working with legacy models in preparing for exploratory analysis.

4.4 Multiresolution, Multiperspective Modeling

4.4.1 Definition

Multi-resolution modeling (MRM) is building a single model, a family of models, or both to describe the same phenomena at different levels of resolution, *and* to allow users to input parameters at those different levels depending on their needs. Variables at level n are abstractions of variables at level n+1. MRM is sometimes called variable- or selectable-resolution modeling. Figure 5 illustrates MRM schematically. It indicates that a higher level model (Model A) itself has more than one level of resolution. It can be used with either two or four inputs. However, in addition to its own MRM features, it has input variables that can either be specified directly or determined from the outputs of separate higher-resolution models (models B and C, shown as "on the side," for use when needed. In principle, one could attach models B and C in the software itself—creating a bigger model. However, in practice there are tradeoffs between doing that or keeping the more detailed models separate. For larger models and simulations, a combination single-model/family-of-models approach is desirable. This balances needs for analytical agility and complexity management.

MRM is not sufficient by itself because of the need for different abstractions or perspectives in different applications. That is, different perspectives—analogous to alternative representations in physics—are legitimate and important. They vary by conception of the system and choice of variables. Designing for both multiple resolution and multiple perspectives can be called MRMPM (pronounced Mr. MIPM).

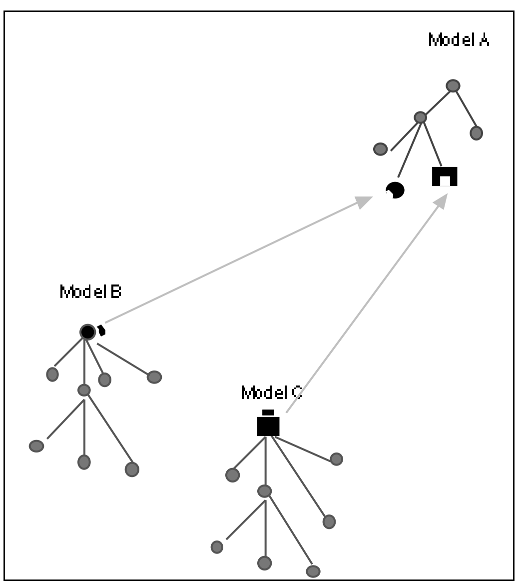


Figure 5: Figure 5: A Multiresolution Family

4.4.2 Mutual Calibration Within a Model Family

Given MRMPM models or families, we want to be able to reconcile the concepts and predictions among levels and perspectives. It is often assumed that the correct way to do this is to calibrate upward: treating the information of the most detailed model as correct and using it to calibrate the higher-level models. This is often appropriate, but the fact is that the more detailed models almost always have omissions and shortcomings. Further, different models of a family draw upon different sources of information—ranging from doctrine or even "lore" on one extreme to physical measurements on a test range at the other.

Figure 6 makes the point that members of a multiresolution model family should be *mutually* calibrated (National Research Council, 1997). For example, we may use low-resolution historical attrition or movement rates to help calibrate more detailed models predicting attrition and movement. This is not straightforward and is often done crudely by applying an overall scaling factor (fudge factor), rather than correcting the more atomic features of the detailed model, but it is likely familiar to readers. On the other hand, much calibration is indeed upward. For example, a combat model with attrition coefficients should typically have adjustments of those coefficients for different circumstances identified in a more detailed model.

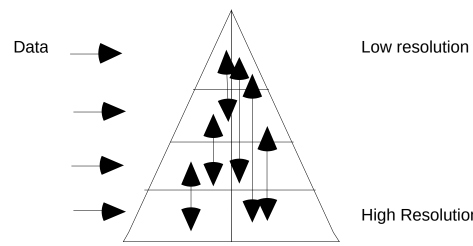


Figure 6: Mutual Calibration of Models in a Family

4.4.3 Design Considerations

So, given their desirability, how do we build a family of models? Or, given pre-existing models, how do we sketch out how they "should" relate before connecting them as software or using them for mutual calibration? Some highlights are as follows.

The first design principle is to recognize that there are limits to how well lower-resolution models can be consistent with high-resolution models. *Approximation is a central concept from the outset.* Several points are especially important:

- Consistency between two models should be assessed in the context of use. What matters is not whether they generate the same final state of the system, but whether they generate approximately the same results in the application (e.g., rank ordering of alternatives). This ties into the well-known concept of experimental frames (Zeigler, et al., 2000).
- Consistency of aggregated and disaggregated models must also be judged recognizing that low-resolution

models may reflect aggregate-level knowledge not contained in the detailed model.

- Comprehensive MRM is very difficult or impossible for complex M&S, but having even some MRM can be far more useful than having none at all.
- Members of an MRM family will typically be valid for only portions of the system's state space. Parameter values (and even functional forms) should change with region.
- Mechanisms are therefore needed to recognize different situations and shift models. In simulations, human intervention is one mechanism; agent-based modeling is another.
- Valid MRM will often require stochastic variables represented by probability distributions. Further, valid aggregate models must sometimes reflect correlations among variables that might naively be seen as probabilistically independent.

With these observations, the ideal for MRM is a hierarchical design for each MRM process, as indicated in Figure 5.

4.4.4 Desirable Design Attributes

From the considerations we have sketched above, it follows that models and analysis methodologies for exploratory analysis should have a number of characteristics. First, they should be able to reflect hierarchical decomposition through multiple levels of resolution and from alternative perspectives representing different “aspects” of a system.

Less obviously, they should also include realistic mechanisms for the natural entities of the system to act, react, adapt, mutate, and change. These mechanisms should reflect the relative “fitness” of the original and emerging entities for the environment in which they are operating. Many techniques are applicable here, including game-theoretic methods and others that may be relatively familiar to readers. However, the most fruitful new approaches are those typically associated with the term agent-based modeling. These include submodels that act “as the agents for” political leaders and military commanders or—at the other extreme— infantry privates on the battlefield or drivers of automobiles on the highway. In practice, such models need not be exotic: they may correspond to some relatively simple heuristic decision rules or to some well-known (though perhaps complex) operations-research algorithm. But to have such decision models is quite different from depending on scripts.

Because it is implausible that closed computer models will be able to meet the above challenge in the foreseeable future, the family of “models” should allow for human interaction—whether in human-only seminar games, small-scale model-supported human gaming, or distributed interactive simulation. This runs against the grain of much common practice.

4.4.5 Stochastic Inputs To Higher Level Models

The last item in the above list is often ignored in today’s day-to-day work. Indeed, too often models that need to be stochastic are deterministic, with quantitatively serious consequences (Lucas, 2000). Often, workers calibrate a high-level (aggregate) model using average outcomes of allegedly "representative" high-resolution scenarios. For example, a theater-level model's air model might be calibrated to results of detailed air-to-air simulation with Brawler, which treats individual engagement classes (e.g., 1 on 1, 1 on 2, … 4 on 8). This may appear to establish the validity of the theater-level model, but in fact the calibration is treacherous. After all, what kinds of engagements occur may be a sensitive function of the sides' command and control systems, strategies, and weather. The calibrations really need to be accomplished on a highly study-specific basis.

Furthermore, the higher-level model inputs often need to be stochastic. Figure 7 illustrates the concept schematically for a simple problem. Suppose that a process (e.g., one computing the losses to aircraft in air-to-air encounters) depends on X,Y, S, and W. But suppose that the outcome of ultimate interest involves many instances of that process with different values of S and W (e.g., different per-engagement numbers of Red and Blue aircraft). An abstraction of the model might depend only on X,Y, and Z (e.g., overall attrition might depend on only numbers of Red and Blue aircraft, their relative quality, and some command and control factor). If the abstraction shown is to be valid, the variable Z should be consistent with the higher-resolution results. However, if it does not depend explicitly on S and W, then there are “hidden variables” in the problem and Z may appear to be a random variable, in which case so also would the predicted outcome F be a random variable. One could ignore this randomness if the distribution were narrow enough, but it might not be.

In the past, such calibrations have been rare because analysts have lacked both theory and tools for doing things better. The “theory” part includes not having good descriptions of how the detailed model should relate to the simplified one. The tool part includes the problem of being able to define the set of runs that should be done (representing the integral of Figure 7) and then actually making those runs.

Ideally, such a calibration would be dynamic within a simulation. Moreover, it would be easy to adjust the calibration to represent different assumptions about command, control, communications, computers, intelligence, surveillance, and reconnaissance (C4ISR), as well as tactics. We are nowhere near that happy situation today,

$$F(X,Y,S,W) \approx \tilde{F}(X,Y,\tilde{Z})$$

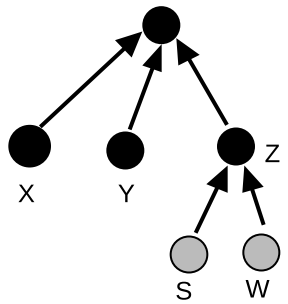


Figure 7: Input to Higher Level Model May Be Stochastic

5 RECENT EXPERIENCE AND CONCLUSIONS

MRMPM is not just idealized theory, but something usable. Over the last several years, my colleagues and I have done considerable work related to the problem of halting an invading army using precision fires from aircraft and missiles. The most recent aspects of that work included understanding in some detail how the effectiveness of such fires are affected by details of terrain, enemy maneuver tactics, certain aspects of command and control, and so on. This provided a good test bed for exploring numerous aspects of MRMPM theory (Davis, Bigelow, and McEver, 2000).

For this work we developed a multiresolution personal-computer model (PEM), written in Analytica, to understand and extend to other circumstances the findings from entity-level simulation of ground maneuver and long-range precision fires. A major part of that work was learning how to inform and calibrate PEM to the entity-level work. There was no possibility, in this instance, of revising the entity-level model. Nor, in practice, did we have such a good understanding of the model as to allow us to construct a comprehensive calibration theory. Instead, we had to construct a new, more abstract, model and attempt to impose some of its abstractions on the data from runs of the entity-level simulation in prior work, plus some special runs made for our purposes. The result is a case history with what are probably some generic lessons learned.

Figure 8 illustrates one aspect of PEM's design. It shows the data flow within a PEM module that generates the impact time (relative to the ideal impact time) for a salvo of precision weapons aimed at a packet of armored fighting vehicles observed by surveillance assets at an earlier time. Other parts of PEM combine information about packet location versus time and salvo effectiveness for targets that happen to be within the salvo's "footprint" at the time of impact, to estimate effectiveness of precision weapons. For the salvo-impact-time module, Figure 8 shows how PEM is designed to accept inputs as detailed as whether there is enroute retargeting of weapons, the latency time, and weapon flight time. However, it can also accept more aggregate inputs such as time from last update. If the input variable Resolution of Time of Last Update Calculation is set "low," then Time From Last Update is specified directly as input; if not, it is calculated from the lower-level inputs.

This design has proven very useful—both for analysis itself and for communicating insights to decision makers in different communities ranging from the C4ISR community to the programming and analysis community. In particular, the work clarified how the technology-intensive work of the C4ISR acquisition community relates to higher-level strategy problems and analysis of such problems at the theater level.

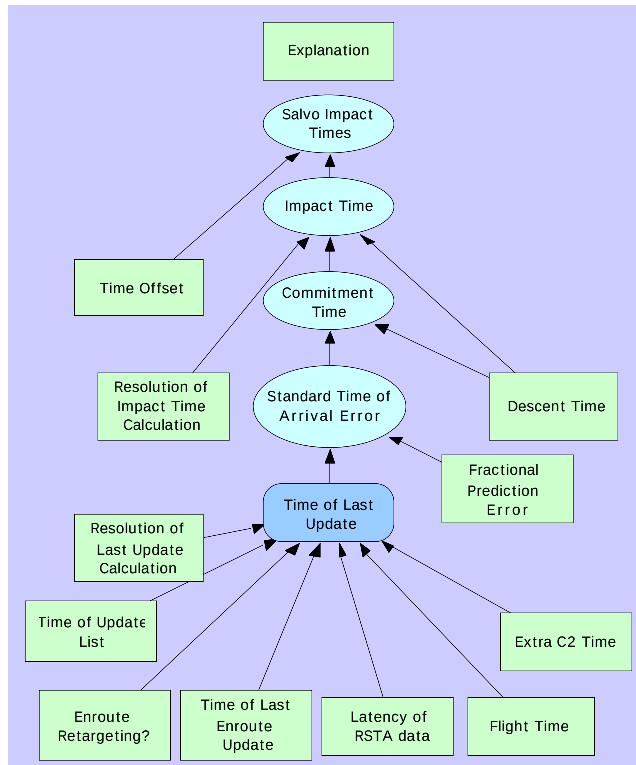


Figure 8: Multiresolution, Multiperspective Design

In other reports (McEver, Davis, and Bigelow, 2000a,b), we describe a broader but more abstract model (EXHALT) that we use for theater-level halt-problem analysis and experiments to deal with the multi-perspective problem. One conclusion is that MRMPM work rather demands a building-block approach that empasizes study-specific assembly of the precise model needed. Although we had some success in developing a closed MRMPM model with alternative user modes representing different demands for resolution and perspective (e.g., the switches in Figure 8), it proved impossible to do very much in that regard: the number of interesting user modes and resolution combinations simply precludes being able to wire in all the relevant user modes. Moreover, that explosion of complexity occurs very quickly. At-the-time-assembly from building blocks, not prior definition, is the stronger approach. This was as we expected, but even more so.

Fortunately, we were able to construct the models needed quickly—in hours rather than days or weeks—as the result of our building-block approach, visual modeling, use of array mathematics, and strong, modular, design.

We also concluded that current personal computer tools—as powerful as they are in comparison with those in past years—are not yet up to the challenge of making the building-block/assembly approach rigorous, understandable, controllable, and reproducible without unrealistically high levels of modeler/analyst discipline. Thus, there are good challenges ahead for the enabling–technology community. Also, the search models for advanced exploratory analysis are not yet well developed.

REFERENCES

Bankes, S. C. 1993. "Exploratory Modeling for Policy Analysis," *Operations Research*, Vol. 41, No. 3.

Bigelow, J. H., P. K. Davis, and J. McEver. 2000. "Case History of Using Entity-Level Simulation as Imperfect Experimental Data for Informing and Calibrating Simpler Analytical Models for Interdiction," *Proceedings of the SPIE*, Vol. 4026.

Davis, P. K (ed.). 1994. *New Challenges in Defense Planning: Rethinking How Much Is Enough*, RAND, Santa Monica, CA.

Davis, P.K.. 2000. "Multiresolution, Multiperspective Modeling as an Enabler of Exploratory Analysis," *Proceedings of the SPIE*, Vol. 4026.

Davis, P. K. and R. Hillestad. 2001. *Exploratory Analysis for Strategy Problems With Massive Uncertainty*. RAND, Santa Monica, CA.

Davis, P.K. and J. H. Bigelow. 1998. *Experiments in Multiresolution Modeling*, RAND, Santa Monica, CA.

Davis, P.K. and M. Carrillo. 1997. *Exploratory Analysis of the Halt Problem: A Briefing on Methods and Initial Insights*, RAND, DB-232, Santa Monica, CA.

Davis, P. K., D. Gompert, R. Hillestad, and S. Johnson, *Transforming the Force: Suggestions for DoD Strategy*, RAND Issue Paper, Santa Monica, CA, 1998.

Davis, P. K., D. Gompert,. and R. Kugler, *Adaptiveness in National Defense: the Basis of a New Framework*, RAND Issue Paper, Santa Monica, CA, 1996.

Davis, P.K., J. H. Bigelow, and J.McEver. 2000. *Effects of Terrain, Maneuver, Tactics, and C4ISR on the Effectiveness of Long Range Precision Fire*, RAND, Santa Monica, CA.

Davis,P.K., J. H. Bigelow, and J. McEver. 1999. *Analytic Methods for Studies and Experiments on Transforming the Force*, RAND, Santa Monica, CA.

Lucas, Thomas. 2000. "The Stochastic Versus Deterministic Argument for Combat Simulations: Tales of When the Average Won't Do," *Military Operations Research*, Vol. 5, No. 3.

McEver,J., P. K. Davis, and J. H. Bigelow. 2000a. *EXHALT: an Interdiction Model for the Halt Phase of Armored Invasions RAND*, Santa Monica, CA.

McEver, J., P. K. Davis, and J.H. Bigelow. 2000b. "Implementing Multiresolution Models and Families of Models: From Entity Level Simulation to Personal-Computer Stochastic Models and Simple 'Repro Models'", *Proceedings of the SPIE*, Vol. 4026.

Morgan, G., and M. Henrion. 1992. *Uncertainty: A Guide to Dealing with Uncertainty in Quantitative Risk and Policy Analysis*, Cambridge University Press, Cambridge, Mass.

National Research Council. 1997. *Modeling and Simulation, Volume 9 of Technology for the United States Navy and Marine Corps*, 2000–2035, National Academy Press, Washington, D.C., 1997.

R. Lempert, .M. E. Schlesinger, and S. C. Bankes. 1996. "When We Don't Know the Costs or the Benefits: Adaptive Strategies for Abating Climate Change," *Climatic Change*, Vol. 33, No. 2.

Zeigler, B. P.,T. G. Kim, and H. Praehofer. 2000. *Theory of Modeling and Design* , Academic Press, San Diego.

AUTHOR BIOGRAPHY

PAUL K. DAVIS is a senior scientist and Research Leader at RAND, and a Professor of Public Policy in the RAND Graduate School. He holds a B.S. from the U. of Michigan and a Ph.D. in Chemical Physics from the Massachusetts Institute of Technology. Dr. Davis is a recipient of the Wanner Memorial Award of the Military Operations Research Society. He is a member of the Naval Studies Board under the National Research Council and has served on a number of studies for the Council, the Defense Science Board, and the National Institute for Standards and Technology. His e-mail and web addresses are pdavis@rand.org and www.rand.org/personal/pdavis.

Position Paper on

DEFINITION AND MEASUREMENT OF MACHINE INTELLIGENCE

By

GEORGE N. SARIDIS

Professor Emeritus RPI

1. DEFINITIONS OF MACHINE INTELLIGENCE

Recently, there have been a lot of arguments on the subject of Intelligence for Machines that operate autonomously and knowledgeably in unfamiliar or hazardous environments. A position view is presented herein, that represents the engineering point of view as the so called "Intelligent Machines" that implement it are designed and built by engineers
(Task Force of the Control System Society of IEEE, chaired by P. Antsaklis1993}

In the last twenty or so years, a lot of discussions have taken place regarding the meaning of **Intelligence**. The psychologists argue about **human intelligence** to be used as the model, while the computer scientists suggest **artificial intelligence** for the job. All these arguments are based on the intelligence that humans demonstrate in dealing with their every day activities, a concept that is still nebulous and very little understood. The engineers stress the concept **of machine intelligence**.

Human Intelligence is to general and poorly understood to be used as model for Intelligent Machines. Artificial Intelligence, on the other hand, was created to deal with the effort to make computers act like human beings, when making decisions and perform other human like activities (Winston 1977). Finally, engineers developed the concept of Machine Intelligence to represent the properties of autonomous machines created to perform unsupervised anthropomorphic tasks (Saridis 1977).

The theory of Intelligent Machines may be thought of as the result of the intersection of the three major disciplines:

- **Artificial Intelligence**,
- **Operations Research**,
- **Control Theory**.

The reason for this claim has been proven necessary is that none of the above disciplines can produce individually a satisfactory theory for the design of such machines. It is also aimed in establishing Intelligent Controls as an engineering discipline, with the purpose of designing Intelligent Autonomous Systems of the future. It combines effectively the results of cognitive systems research, with various mathematical programming control techniques. The control intelligence is hierarchically distributed according to the **Principle of Precision with Decreasing Intelligence (IPDI)**, evident in all hierarchical management systems. The analytic functions of an Intelligent Machine are implemented by Intelligent Controls, using Entropy as a measure. Such an architecture is analytically implemented using entropy as a measure.

However, various cost functions expressed in entropy terms, may be used to evaluate the generated design. Reliability, a property highly desirable for systems functioning autonomously, is a very desirable measure of performance and can also be expressed by entropy, and can be combined in the criterion of performance of the system design.

Intelligence is defined according to the *American Heritage Dictionary (1992) as* :

- Intelligence (Human) is defined as the capacity to acquire and apply Knowledge.

Such a statement implies that knowledge is the key variable in an Intelligent system.

The following two definitions, due to P. H. Winston and G. N. Saridis respectively are given to clarify the subject (Winston 1977).

Artificial Intelligence is represented as a mapping of anthropomorphic tasks into the analytic tools of the computer in order to study human behavior, while Machine Intelligence is the inverse mapping of analytic tools imbedded in a machine into anthropomorphic tasks.

- **Artificial Intelligence is the study of ideas which enable computers to do the things that make people seem intelligent. Its central goals are to make computers more useful and to understand the the principle, which makes Intelligence possible.**

The key components of Artificial Intelligence are: interactive systems between man and machine, heuristics and expert system exaustive programming (Saridis 1977).

Some definitions regarding Machine Knowledge and Intelligence are appropriate in order to clearly define the field of Intelligent Machines. According to the *American Heritage Dictionary (1992)* :

- Intelligence is defined as the capacity to acquire and apply Knowledge.

Such a statement implies that knowledge is the key variable in an Intelligent system. Since we shall be dealing with **Machine Intelligence** an appropriate definition is necessary:

- **Machine intelligence is defined as the process of analyzing, organizing and converting data into Machine Knowledge.**

The key components of Machine Intelligence are: computer mathematics, cognitive engineering and Intelligent control.

Now it is well understood that:

- **Knowledge is a form of structured Information.**

This is very convenient because an analytic formulation of Intelligent Machines may be developed, using Shannon's Information Theory. Therfore, **Machine Knowledge** is defined as:

- **Machine knowledge is defined to be structured information acquired and applied to remove ignorance or uncertainty about a specific task pertaining to an Intelligent Machine.**

Similarly,

- **The Rate of Machine Knowledge is the flow of Knowledge in an Intelligent Machine.**

Using the above definitions analytic expressions of Machine Knowledge and its Rate are obtained.

Assuming that Machine Knowledge is Information:

$$K = - \ln[p(K)] \qquad (1.1)$$

and the average Rate of Knowledge is also:

$$R = - a - \mu \ln[p(R)] \qquad (1.2)$$

where $p(\cdot)$ is the probability density of the event. Solving for p(R) we obtain:

$$p(R) = \exp(- a - \mu R) \qquad (1.3)$$

$$a = \ln {}_{Ox} \exp(-\mu R)\ dx$$

Complexity is always imbedded in the design and execution of Intelligent Machines. However, their performance is always prescribed by a certain level of detail required by the task expected to be executed, which is defined as Precision. Such details are inversely associated with the uncertainty of execution and thus are measurable with entropy. The following definitions help to clarify this concept.

- <u>Precision</u> is the compliment of the uncertainty of execution of the various tasks of an Intelligent

Machine, and Imprecision serves as a measure of the complexity of the process.

The concept of precision will therefore be associated with the Principle of Increasing Precision and Decreasing Intelligence. Precision is required for the smooth and accurate execution of tasks associated with world processes.

This generalization is found useful in order to accommodate unconventional systems that are served by Intelligent Machines, like biological, environmental etc (Prigogine 1980).

Intelligent Control is the main tool to implement Intelligent Machines. In order to properly implement the Theory of Intelligent Machines the broader definition of Automatic Control Systems is used.

- Control is making a Process do what we want it to do.

The Theory of Hierarchically Intelligent Controls has been recently reformulated by Saridis(1996) to incorporate new architectures that are using Neural and Petri nets. The analytic functions of Hierarchically Intelligent Machines are implemented using Entropy as a measure. The resulting hierarchical control structure is based on the Principle of Increasing Precision with Decreasing Intelligence (IPDI) which is discussed in the next Chapter. Each of the three levels of the Intelligent Control is using different architectures, in order to satisfy the requirements of the Principle:

The Organization level
modeled after a Boltzmann machine for abstract reasoning, task planning and decision making;

The Coordination level
composed of a number of Petri Net Transducers supervised by a dispatcher for command management, serving as an interface with the Organization level;

The Execution level,
includes the sensory, navigation and control hardware which interacts one-to-one with the appropriate Coordinators, while a VME bus provides a channel for database exchange among the several devices.

This system was implemented on a robotic tele-transporter, designed for construction of trusses for the Space Station

Freedom, at the Center for Intelligent Robotic Systems for Space Exploration laboratories at the Rensselaer Polytechnic Institute.

The basic concepts underlining the theory of Hierarchically Intelligent Machines like Machine Intelligence, Machine Knowledge, Precision and Complexity were defined and contrasted to Artificial and Human Intelligence. The basic difference being the search for an analytic formulation that would lead to an engineering implementation. Further more it has been recently realized that other scientific disciplines have being using the same concepts for an analytic representation of their subjects (Prigogine 1980). Such ideas will be discussed in the next Chapter.

2. THE ENTROPY CONCEPT

Entropy is a form of lower quality energy, first encountered in Thermodynamics. It represents an undesirable form of energy that is accumulated when any type of work is generated. Recently it served as a model of different types of energy based resources, like transmission of information, biological growth, environmental waste, etc. Entropy was currently introduced, as a unifying measure of performance of the different levels of an Intelligent Machine by Saridis (1985). Such a machine is aimed at the creation of **modern intelligent machines which may perform human tasks with minimum interaction with a human operator**. Since the activities of such a machine are energy related, entropy may easily serve as a cost measure of performing various tasks as Intelligent Control, Image Processing, Task Planning and Organization, and System Communication among diversified disciplines with different performance criteria. The model to be used is borrowed from Information Theory, where the uncertainty of design is measured by a probability density function over the appropriate space, generated by **Jaynes' Maximum Entropy Principle.**

Other applications of the Entropy concept are for defining Reliability measures for design purposes and obtaining measures of complexity of the performance of a system, useful in the development of the theory of Intelligent Machines.

Entropy is a convenient global measure of performance because of its wide applicability to a large variety of systems of diverse disciplines including waste processing, environmental, socio-economic, biological and other. Thus, by serving as a common measure, it may expand system integration by incorporating say societal, economic or even environmental systems to engineering processes.

The concept of **Entropy** was introduced **in Thermodynamics** by Clausius in 1867, as the low quality energy resulting from the second law of Thermodynamics. This is the kind of energy which is generated as the result of any thermal activity, at the lower thermal level, and is not utilized by the process.

It was in 1872, though, that Boltzmann used this concept to create his **theory of statistical**

thermodynamics, thus expressing the uncertainty of the state of the molecules of a perfect gas. The idea was created by the inability of the dynamic theory to account for all the collisions of the molecules, which generate the thermal energy. Boltzmann (1872) stated that the entropy of a perfect gas, changing states isothermally, at temperature T is given by;

$$S = - k \int_X (? -H)/kT \exp\{(? -H)/kT\}\, dx \qquad (2.1)$$

where ? is the Gibbs energy, ? = - kT ln exp {-H/kT}, H is the total energy of the system, and k is Boltzmann's universal constant. Due to the size of the problem and the uncertainties involved in describing its dynamic behavior, a probabilistic model was assumed where the Entropy is a measure of the molecular distribution. If p(x) is defined as the probability of a molecule being in state x, thus assuming that,

$$p(x) = \exp\{(? -H)/kT\} \qquad (2.2)$$

where p(x) must satisfy the "incompressibility" property over time, of the probabilities, in the state space X, e.g.;

$$dp/dt = 0 \qquad (2.3)$$

The incompressibility property is a differential constraint when the states are defined in a continuum, which in the case of perfect gases yields the Liouville equation. Substituting eq.(2.2) into eq.(2.1) the Entropy of the system takes the form,

$$S = - k \int_X p(x) \ln p(x)\, dx \qquad (2.4)$$

The above equation defines Entropy as a measure of the uncertainty about the state of the system, expressed by the probability density exponential function of the associated energy.

Actually, the problem of describing the entropy of an isothermal process should be derived from the Dynamical Theory of Thermodynamics, considering heat as the result of the kinetic and potential energies of molecular motion. It is the analogy of the two formulations that led into the study of the equivalence of entropy with the performance measure of a control system. If the Dynamical Theory of Thermodynamics is applied on the aggregate of the molecules of a perfect gas, an Average Lagrangian I, should be defined to describe the average performance over time of the state x of the gas,

$$I = \int_{t0}^{tf} L(x,t)\, dt \qquad (2.5)$$

where the Lagrangian L(x,t) = (Kinetic energy) - (Potential energy). The Average Lagrangian when minimized, satisfies the **Second Law of Thermodynamics**. Since the formulations eqs.(2.1) and (2.5) are equivalent, the following relation should be true;

$$S = I/T \qquad (2.6)$$

where T is the constant temperature of the isothermal process of a perfect gas (Lindsay and Margenau, 1957). This relation will be the key in order to express the performance measure of the control problem as Entropy.

In the 1940's Shannon (1963), using Boltzmann's idea, e.g., eq. (3.4), defined **Entropy** (negative) as a measure of the **uncertainty of the transmission of information**, in his celebrated work on **Information Theory**:

$$H = - \int_{\Omega} p(s) \ln p(s)\, ds \qquad (2.7)$$

where p(s) is a Gaussian density function over the space Ω of the information signals transmitted. The similarity of the two formulations is obvious, where the uncertainty about the state of the system is expressed by an exponential density function of the energy involved.

Shannon's theory was generalized for dynamic systems by Ashby (1965), Boettcher and Levis (1983), and Conant (1976) who also introduced various laws which cover information systems, like the Partition Law of Information rates.

The **e-entropy** formulation of the metric theory of complexity, originated by Kolmogorov (1956) and applied to system theory by Zames (1979) is another use of entropy.

It implies that an increase in knowledge about a system, decreases the amount of e-entropy which measures the uncertainty (complexity) involved with the system.

$$e - H = \ln(n_e) \qquad (2.8)$$

where n_e is the minimum number of coverings of a set e. Therefore e-entropy is a measure of complexity of the system involved. It may also be interpreted as a measure of precision if it viewed as the number of points required to describe a line.

Since the latest major improvements in the average quality of life, major increases have occurred in the production of waste, traffic congestion, biological pollution and in general social and environmental decay(Bailey 1990, Brooks Wiley1988, Prigogine1996, Rifkin 1980), which can be interpreted as the increase of the Global Entropy of our planet (Saridis 1998), an energy that tends to deteriorate the quality of our modern society. According to the second axiom of thermodynamics this is an irreversible phenomenon, and nothing can be done to eliminate it.

In an attempt to generalize the principle used by Boltzmann and Shannon to describe the uncertainty of the performance of a system under a certain operating condition, Jaynes (1957) formulated his **Maximum Entropy Principle**, to apply it in Theoretical Mechanics. In summary it claims that

- **The uncertainty of an unspecified relation of the function of a system is expressed by an exponential density function of a known energy relation associated with the system.**

As an example of the use of Entropy as a measure of performance, is a modified version of the Principle, as it applies to the Control problem, is derived in the sequel, using Calculus of Variations (Saridis 1987). The proposed derivation represents a new formulation of the control problem, either for deterministic or stochastic systems and for optimal or non-optimal solutions.

The purpose of this work is to establish entropy measures, equivalent to the performance criteria of the optimal control problem, while providing a physical meaning to the latter. This is done by expressing the problem of control system design probabilistically and assigning a distribution function representing the uncertainty of selection of the optimal solution over the space of admissible controls. By selecting the worst case distribution, satisfying **Jaynes' Maximum Entropy Principle**, the performance criterion of the control is associated with the entropy of selecting a certain control (Jaynes 1957, Saridis 1985). Minimization of the differential entropy, which is equivalent to the average performance of the system, yields the optimal control solution. Furthermore, the Generalized Hamilton-Jacobi-Bellman equation is derived from the incompressibility over time condition of the probability distribution. Adaptive control and stochastic optimal control are obtained as special cases of the optimal formulation, with the differential entropy of active transmission of information, claimed by Fel'dbaum (1965), as their difference. Upper bounds of the latter may yield measures of goodness of the various stochastic and adaptive control algorithms.
In this section, the entropy measure for optimal control will be established.

The optimal feedback deterministic control problem with accessible states is defined as follows: given the dynamic system;

$$dx/dt = f(x,u,t) \; ; \; x(t_0) = x_0;$$

and the cost function,

$$V(u;x_0,t_0) = \int_{t0}^{T} L(x,u,t)\, dt \qquad (2.9)$$

where $x(t) \in \Omega_x$ is the n-dimensional state vector $u(x,t) \in \Omega_u XT \subset \Omega_x XT$, is the m-dimensional feedback control law and $t \in \Im = [t_0,T]$.

An optimal control $u^*(x,t)$ is sought to minimize the cost,

$$V(u^*;x_0,t_0) = \underset{u}{\text{Min}} \int_{t0} L(x,u,t)\, dt \qquad (2.10)$$

Define the differential entropy, for some u(x,t),

$$H(x_0,u(x,t),p(u)) = H(u) = -\int_{\Omega x0}\int_{\Omega x} p(x_0,u)\ln p(x_0,u)\, du dx_0 \qquad (3.11)$$

where $x_0 \in \Omega_{x0}$, $x \in \Omega_x$ the spaces of initial conditions and states respectively, and $p(x_0,u)=p(u)$ the probability density of selecting u. One may select the density function p(u) to maximize the differential entropy according to **Jaynes' Maximum Entropy Principle** (Jaynes 1957), subject to $E\{V(x_0,u,t)\}=K$, for some u(x,t). This represents a problem more general than the optimal where K is a fixed but unknown constant, depending on the selection of u(x,t).

For appropriate constants ? and μ, the worst case density is,

$$p(u) = \mathbf{e}^{-?-\mu V(u(x,t),x0,t0)}$$

$$e^{?} = \int_{\Omega x} e^{-\mu V(u(x,t),x0,t0)}\, dx \qquad (2.12)$$

and the total Entropy is equivalent to the average cost function:

$$H(u) = ? + \mu E\{V(u(x,t),x_0,t_0)\} \qquad (2.13)$$

and the corresponding minimum value with respect to u(x,t) represents the optimal design.

In most organization systems, the control intelligence is hierarchically distributed from the highest level which represents the most intelligent manager to the lowest level which represents the worker, which is a manifestation of the Principle of **Increasing Precision with Decreasing Intelligence (IPDI)**,. On the other hand, the precision (complexity) or skill of execution is distributed in an inverse manner from the bottom to the top as required for the most efficient performance of such complex systems. This has been analytically formulated as the **Principle of Increasing Precision with Decreasing Intelligence (IPDI)**, by Saridis (1989). The formulation and proof of the principle is based on the concept of Entropy in that report.

According to the IPDI Principle:

- **Machine Intelligence (MI) is the set of actions and/or rules which operates on a Data-base (DB) of events or activities to produce flow of knowledge.**

$$\textbf{(MI) : (DB)} \Rightarrow \textbf{(R)} \qquad (2.14)$$

This principle suggests that for constant flow of knowledge through the machine less intelligence more data (complexity) are required. Thus it provides an interesting definition of **Machine Intelligence** that has been debated. This has been realized in the three level architecture of Intelligent Machines discussed in the previous section.

3. CONCLUSIONS

A set of definitions leading to the concept of **Machine Intelligence** have been discussed in this paper, and it is contrasted to **Artificial and Human Intelligence**. Entropy, defined as a Universal Energy, resulting from the production of Work in a system, may successfully serve as a measure of **Machine Intelligence**. The production of Entropy is irreversible without the use of additional work, and may represent thermal energy in Thermodynamics, Information in Communication systems, Performance in Control systems, as well as waste and pollution in Ecological systems, Economic spending in Societal systems, or Biodegradation in Biological systems (Bailey 1990, Boltzmann 1872, Brooks Wiley1988, Prigogine1996, Shannon 1963, Saridis 1998, Rifkin 1980). It represents an unifying measure for globalization of many, up to now, disjoint sciences and may successfully be used as measure for the development of Hierarchically Intelligent Machines with the use of the **Principle of Increasing Precision with Decreasing Intelligence.**

REFERENCES

Antsaklis, P., et al (1993), *Final Report of the Task Force on Intelligent Control* *IEEE Control Systems Society Magazine* December.

Ashby W. R., (1975), ***An Introduction to Cybernetics***,J. Wiley & Sons, Science Edition, New York.

Bailey K. D., (1990), ***Social Entropy Theory,*** State University of New York Press, Albany N.Y.

Boettcher K. L., Levis A. H., (1983),"Modeling the Interacting Decision-Maker with Bounded Rationality", *IEEE Transactions on System Man and Cybernetics*, ***Vol. SMC-12***, 3, pp. .

Boltzmann L. (1872), "Further Studies on Thermal Equilibrium Between Gas Molecules", Wien Ber., ***Vol. 66***, p. 275.

Brooks D. R. and Wiley, E. O. (1988) ***Evolution as Entropy*** University of Chicago Press, Chicago Il.

Conant, R. C., (1976), "Laws of Information which Govern Systems", *IEEE Transactions on System Man and Cybernetics*, ***Vol. SMC-6***, No. 4, pp. 240-255.

Faber M., Niemes N., Stephan G., (1995), ***Entropy, Environment and Resourses,*** Springer-Verlag Berlin Germany.

Feld'baum, A.A. (1965), ***Optimal Control Systems***, Academic Press, New York.

Jaynes, E.T. (1957), "Information Theory and Statistical Mechanics", *Physical Review*, ***Vol.4***, pp. 106.

Kolmogorov, A.N. (1956), "On Some Asymptotic Characteristics of Completely Bounded Metric Systems", *Dokl Akad Nauk*, SSSR, ***Vol. 108***, No. 3, pp. 385-389.

Lindsay, R.B., Margenau, (1957), ***Foundations of Physics***, Dover Publications, New York NY.

McInroy J.E., Saridis G.N.,(1991), "Reliability Based Control and Sensing Design for Intelligent Machines", in ***Reliability Analysis*** ed. J.H. Graham, Elsevier North Holland, N.Y.

Prigogine, I., (1980), ***From Being to Becoming***, W. H. Freeman and Co. San Francisco, CA.

Prigogine, Ilya, (1996), *La Fin des Certitudes* Editions Odile Jacob, Paris France.

Rifkin, Jeremy, (1989) *?ntropy into the Greenhouse World* Bantam Books New York.

Saridis, G.N. (1979), "Toward the Realization of Intelligent Controls", *IEEE Proceedings*, ***Vol. 67***, No. 8.

Saridis, G. N. (1983), "Intelligent Robotic Control", *IEEE Trans. on Automatic Control*, ***Vol. 28***, No. 4, pp. 547-557, April.

Saridis, G.N. (1985), "An Integrated Theory of Intelligent Machines by Expressing the Control Performance as an Entropy", *Control Theory and Advanced Technology*, ***Vol. 1***, No. 2, pp. 125-138, Aug.

Saridis, G.N. (1988), "Entropy Formulation for Optimal and Adaptive Control", *IEEE Transactions on Automatic Control*, ***Vol. 33,*** No. 8, pp. 713-721, Aug.

Saridis, G.N. (1989), "Analytic Formulation of the IPDI for Intelligent Machines", ***AUTOMATICA the IFAC Journal, 25***, No. 3, pp. 461-467.

Saridis, G. ?. (1995) *Stochastic ?rocesses, ?stimation, and Control: ?he ?ntropy ?pproach*, John Wiley and Sons, ?ew York.

Saridis, G.N., (1996)," Architectures for Intelligent Controls" *Chapter 6*, in ***Intelligent Control Systems***, M. M. Gupta, N. K. Singh (eds) IEEE Press New York NY.

Saridis, G. N., (1998), "Optimal Control of Global Entropy for Environmental Systems"***IEEE Robotics and Automation Magazine***, Vol. 5, No. 3, Sept.

Saridis, G.N. and Graham, J.H. (1984), "Linguistic Decision Schemata for Intelligent Robots", ***AUTOMATICA the IFAC Journal, 20***, No. 1, pp. 121-126, Jan.

Shannon, C. and Weaver, W. (1963), *The Mathematical Theory of Communications*, Illini Books.

Valavanis, K.P.,Saridis, G.N., (1992), ***Intelligent Robotic Systems: Theory and Applications,*** Kluwer Academic Publishers, Boston MA.

Wang, F., Saridis, G.N. (1990) "A Coordination Theory for Intelligent Machines" ***AUTOMATICA the IFAC Journal, 35,*** No. 5, pp. 833-844,Sept.

Winston P. H. (1984), ***Artificial Intelligence,*** Addison Wesley, Reading MA.

Zames, G. (1979), "On the Metric Complexity of Casual Linear Systems, $\in$-entropy and $\in$-dimension for Continuous Time", *IEEE Trans. Autom. Control, 24*, 2, pp. 220-230, April.

Domain Independent Measures of Intelligent Control

David Friedlander
Shashi Phoha
Applied Research Laboratory
The Pennsylvania State University
University Park, PA 16802

Asok Ray
Mechanical Engineering Department
The Pennsylvania State University
University Park, PA 16802

ABSTRACT

There is no standard method for measuring intelligence in artificial systems. One reason for this is that no single definition of intelligence exists. Another is that many of the definitions of intelligence are not appropriate for artificial systems based on the current level of scientific understanding. This includes *introspective*, as opposed to *behavioral*, measurements. This paper explores quantitative, domain independent measures of intelligence for discrete event control systems. It is motivated by traditional measures of effective control such as *controllability*, and *robustness*, and includes original work on *robustness* and *permissiveness*.

KEYWORDS: *intelligence, artificial systems, control systems, hierarchical control, intelligent control*

1. INTRODUCTION

1.1. INTELLIGENCE OF ARTIFICIAL SYSTEMS

A single Intelligence Quotient, as generated by a standardized IQ test, has been used as a measure of general human intelligence. More recent work suggests that there are multiple types of intelligence. This concept is essential to the development of intelligence measurements for artificial systems for two reasons. First, current systems have not reached the level where they can display behavior indicative of a nontrivial understanding of the general representations of human knowledge such as language or mathematics. Second, current techniques in constructing artificial systems result in a sharp trade-off between the quality of performance and the breadth of the domain. In order to perform at a reasonable level, most artificial systems are restricted to a relatively narrow domain.

The current criteria for intelligent systems tend to look at either how well the system performs its assigned tasks, or to what extent can the system behave in ways that are characteristic of human intelligence. The former criteria tend to be domain specific, implementation independent, and relatively easy to quantify and measure. The latter criteria tend to be domain independent, implementation dependent, and are difficult to quantify and measure.

Domain independent intelligent behaviors that can be approximated in current artificial systems include:

- reacting effectively to novel stimuli and situations,
- perceiving essential properties from a large, complex world of sensory information,
- identifying and taking action on the essential problem of a given situation,
- making appropriate decisions in a variety of situations, in complex environments,
- recognizing and exploiting opportunities within one's environment,
- recognizing patterns within the environment,
- manipulating symbols,
- overcoming obstacles,
- correcting for errors,
- handling uncertainty,
- and learning from experience.

These behaviors are difficult to measure in the general case. This paper presents quantitative measurements of intelligent behaviors for systems based on hierarchical networks of discrete event controllers. Some of the metrics used here are based on extending methods from continuous control to discrete event control systems.

The methods are illustrated with two types of examples. The first is a highly simplified control system for command and control (C^2) of aircraft operations in battle management as illustrated in Figure 4. The second is an abstract controller of slightly greater complexity. The lowest level is shown in Figure 1; hierarchical versions are shown in Figures 2 and 3.

This paper examines the following characteristics of hierarchical control systems:

- *Controllability*, the ability of the system to accomplish its goals without reaching an error state from which it cannot recover.
- *Hierarchical Consistency*, the ability of a higher-level controller to achieve its goals indirectly, by controlling one or more lower-level controllers.
- *Robustness*, the ability of the system to operate under uncertainty and novel situations, and to recover from errors,

- *Permissiveness*, the ability to achieve goals through more than one path.
- *Aggregation*, the ability to abstract essential properties from lower level data,
- *Disaggregation*, the ability to take effective specific actions based on higher level abstractions,
- *Scalability*, the ability to handle large, complex environments, and

1.2 HIERARCHICAL FINITE STATE AUTOMATON (FSA) CONTROL SYSTEMS

Although, computational theory shows that finite state automata (FSA) are more restricted than more general representations such as general finite state machines, it has been shown that networks of FSAs do have the same computational power as general finite state machines [5] such as digital computers.

Control systems have a history of successfully operating large complex systems of interacting components in real-time. For most of this history, control systems were *continuous*. They used sensors to collect data from various points in the system (often referred to as the *plant*), process the data, and perform actions on the plant through the use of mechanical devices. A number of metrics such as controllability and robustness have been developed to measure the performance of the control system in controlling the plant.

The concept of a continuous control system was extended to discrete event systems (DES) by Ramadge and Wonham [7]. Instead of processing continuous numerical data, DES controllers process strings of symbols that form a formal language. This has resulted in a synergy with work on processing formal languages from the discipline of Computer Science, and resulted in mathematical theorems that provide ample insight to the strengths and limitations of the approach. Artificial Intelligence approaches do not have the same extent of mathematical grounding.

In DES control, the plant can be considered as a machine that processes symbols from a (finite) alphabet in a way that forms a formal language, L, over an alphabet of symbols, Σ. The alphabet, Σ, can be partitioned into symbols corresponding to uncontrollable events, Σ_u, and symbols corresponding to controllable events, Σ_c. These symbols represent events in the system. An example of a controllable event is a friendly (controlled) aircraft firing at an enemy target, while an example of an uncontrollable event is the enemy target firing on the friendly aircraft. The controller can be thought of as a recognizer of uncontrollable events and a generator of controllable ones. In analogy to continuous control systems, the controllable events are the plant input (feedback), and the uncontrollable events are the plant output.

The association of DES controller actions with decisions expands the notion of intelligent control towards general intelligence. Performance measurements from continuous control theory have been extended to include DES control. New measurements, which are mathematically grounded, have also been developed for DES controller performance. DES control systems can exhibit intelligent behavior in the same way as AI software or robots. The performance measures for intelligent DES control are an indication of intelligence in the more generic sense.

More recent work has extended the notion of single DES controllers to interacting hierarchies of DES controllers. This includes mathematical work to extend the notion of controllability to hierarchies, where it is called *hierarchical consistency* [13]. The use of a hierarchy allows the controller to make complex decisions while taming the explosion of the number of states that would take place in a single controller performing the same function. Lower level controllers transmit an *aggregated* version of their formal languages to higher-level controller nodes. The higher-level nodes exert control on lower levels by enabling and disabling controllable events at the lower levels. Such systems exhibit behavior analogous to forming conclusions and formulating and executing plans.

2. CONTROLLABILITY AND HIERARCHICAL CONSISTENCY

The controllable events of the system can be disabled, i.e., prevented from occurring, by the controller. It uses this ability to influence the evolution of the system by preventing certain events from occurring at certain times.

The goals of the control system are given as a set of specifications such as, "If the aircraft runs out of weapons, it returns to base." Given the uncontrolled language of the plant, L, the specifications can be formalized as a sublanguage of L, $K \subseteq L$. The controller attempts to enforce the specifications by restricting the plant to operate within K, rather than L. The plant is said to be *controllable* by the controller if it can operate the plant in a way that satisfies the specifications.

A controllable system is able to follow its specifications from any of the system states. Even when an uncontrollable event moves the system in an unanticipated way, there is a path that satisfies the specifications. This shows some characteristics of purposeful behavior.

FSA controllers can be organized in a hierarchy where high-level controllers achieve high-level goals by observing and controlling low-level controllers. *Hierarchical consistency*, the hierarchical equivalent of controllability, is the ability of the high-level controller to achieve its goals in this way, starting from any legal combination of states in the high and low level controllers. This behavior, when it can be achieved, shows the ability of the system to achieve high-level goals through low-level actions, suggesting an ability to formulate and carry out plans.

2.1. CONTROLLABILITY

The controllability definition can be formalized as: K is controllable if $\overline{K}\Sigma_u \cap L \subseteq \overline{K}$, where $\overline{K}$ is the prefix closure language of K. That is, the occurrence of any uncontrollable event at any time permitted by the dynamics of the plant, will not violate the control specifications.

Figure 1 shows how a controller would be implemented in practice. The controller has been implemented as a finite state machine with the specification that the plant should always return to state S_0. The first event, a, moves the controller from state S_0 to state S_1 and the second event, b, moves it from state S_1 to state S_2. At state S_2, the controller takes action, e, and moves to state S_4, where it takes action h, and returns to state S_0. The system's performance is characterized by the string a,c,e,h, which is in $\overline{K}$ if the plant is controllable by the controller.

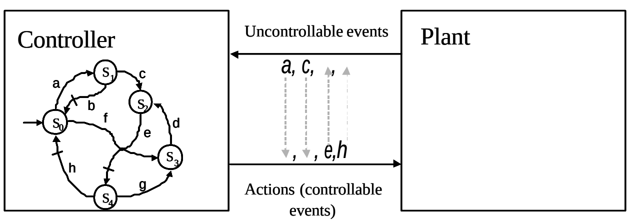


Figure 1. Discrete Event System Control

The example exhibits behavior that looks purposeful. It seeks to bring the plant back to a goal state S_0. It also gives the appearance of overcoming obstacles, uncontrollable events, in an uncertain environment. For example it could not perform action e when it was in state S_1, because event c occurred. It then achieved its goal in another way, by taking actions e and h. If the plant is controllable, there will always be a path to the goal state.

In a large and complex enough system, these behaviors would appear intelligent. These types of systems have been implemented to solve complex applications such as controlling an automated factory.

2.2. HIERARCHICAL CONSISTENCY

A hierarchical structure is used in control of dynamic systems for a variety of tasks. Control is divided between higher levels, which process events of greater generality and larger scope; and lower levels, which process more specific events of lesser scope.

This notion has been formalized in a way that is illustrated in Figure 2. The higher level controller is designed to control a virtual high level plant with the following components: the low-level, i.e. *actual*, plant; the low-level controller, M, a mapping from strings of the low-level plant language to high-level events; and U, a mapping from high-level, controllable events to low-level *control patterns*. A control pattern is a set of low-level controllable events that are to be disabled in the low-level controller.

We start with a low-level plant with language L_p, and a low-level controller with language $K_{lo} \subseteq L_p$. We wish to implement further restrictions on the performance of the low-level plant to a language $\tilde{K}_{lo} \subseteq K_{lo} \subseteq L_p$. This is to be done by translating strings from L_p to a high level language, $L_{hi} = M(L_p)$, which has an alphabet that containing controllable and uncontrollable symbols, $\Sigma^{hi} = \Sigma_u^{hi} \cup \Sigma_c^{hi}$. In the example, each state in the low-level controller is labeled with either a high-level symbol or t_0. This implicitly determines M. Whenever the low-level controller enters a state marked with a high-level symbol, the symbol is added to the high-level translation of the low-level string.

In order for this scheme to work, the high-level controller needs to be able to control the virtual high-level plant via the mechanism shown in Figure 2. This is called *hierarchical consistency*. It is true when $M(\tilde{K}_{lo}) = K_{hi}$.

Note that the example is hierarchically consistent because

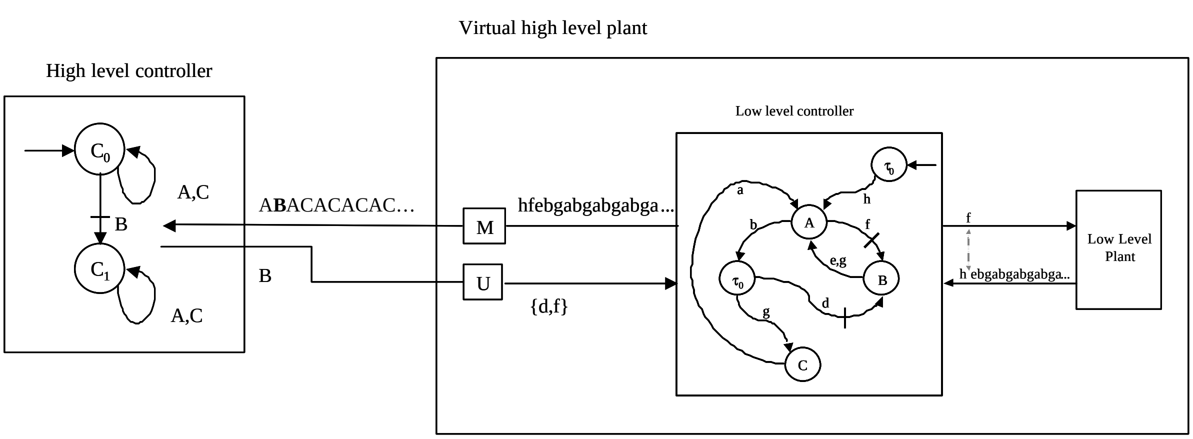


Figure 2. Hierarchical Control

the only controllable high-level event, B, can be disabled with the its associated control pattern {d,f}, which blocks the low-level controller from entering the state which transmits B to the high-level controller. This is done by disabling all of the low-level (controllable) events leading to the low-level state marked B.

The high-level controller adds the following requirement to the behavior of the low-level controller: *an event from the set {d,f} can only occur once in any low-level string*. This translates to the high-level requirement that: *the event* B *can only occur once in any high-level string.*

3. Robustness and Permissiveness

Two additional concepts in DES control theory that relate to intelligent behaviors are *robustness* and *permissiveness*. Robustness is the ability of the controller to handle uncertainty. In this section we will look at robustness with respect to uncertainty in the characteristics of the actual plant. Permissiveness is the ability of the controller to allow a wide range of behaviors while operating the plant within the control specifications. This behavior gives the appearance of resourcefulness. It may allow the system to satisfy its requirements in multiple ways and might improve performance in the real world where a single path to a goal may be blocked by an unanticipated event.

In this section, we first propose a measure for formal languages and then use it to derive quantitative measures of robustness and permissiveness.

3.1. Robustness

In reality, the actual plant is not completely known. It is represented by an FSA, called the nominal plant model, which contains all of the available knowledge about the actual plant. The language of the plant is approximated by the language of the nominal plant model, and the controller is designed on this basis.

It is therefore important to determine whether the controller can control languages (i.e., plants) other that the one for which it was designed. This quantity is known as *robustness*. The more robust the controller, the more likely it is to be able to control the actual plant.

We have determined a method to estimate robustness through the development of two concepts, a population of plant languages near the language of the nominal plant model, and a measure for formal languages of a given alphabet, Σ.

The population of plant languages can be defined as the languages of plants derived from the nominal plant model through the application of a small number of primitive operations such as the addition or deletion of a single state or transition, under the restriction that the results define a deterministic FSA.

The measure of formal languages can be defined in terms of a weighted partition of the set Σ^* of all finite strings in the alphabet Σ. The countable set Σ^* is partitioned into disjoint subsets, S_i, such that $\Sigma^* = \bigcup_{i=1}^{n} S_i$, where n is at most countable infinity (i.e., either n is a finite positive integer or infinity). Each subset is assigned a weight, w_i such that $\sum_1^n w_i = 1$. The measure of a given language L, is defined as:

$$m(L) = \sum_1^n w_i \Delta_i(L)\,, \tag{1}$$

where $\Delta_i(L) = \begin{cases} 1 \; if \; \exists s \in L \; such\, that \; s \in S_i \\ \quad 0 \;\; otherwise \end{cases}$

The robustness of a controller, C, can then be defined as: $R(C) = \sum_{P_i \in R} M(L(P_i) - L(P)) \cdot D(L(P_i))$, where R is the population of plant models related to P, the nominal plant model; and $D(L) = 1$ if C can control L, 0 otherwise.

3.2. Permissiveness

Given a set of specifications in the form of a language, E, a controller, K_1, controls a plant, G, if $L(G \mid K_1) \subseteq E$. There could be, however, another controller, K_2, such that $L(G \mid K_1) \subseteq L(G \mid K_2) \subseteq E$. In this case, K_2 is considered more *permissive* than K_1. Although both controllers control the plant, K_2 allows a greater range of behaviors in the closed loop system.

Since $L(G \mid K) \subseteq L(G)$, for any controller defined on G, it follows that, for any controller K which satisfies E, $L(G \mid K) \subseteq L(G) \cap E$. Using the proposed language measure, we can define the permissiveness of a controller, K, defined on a plant, G, operating within the specification language, E, as

$$P(C) = \left(\frac{m(L(G \mid K))}{m(L(G) \cap E)} \right) \in [0,1]. \tag{2}$$

4. Aggregation, Disaggregation, and Scalability

The technique for hierarchical control, described in Section 2, can be extended to allow a high-level controller to control more than one low-level plant. It can also be extended to form multilevel hierarchies of arbitrary size. The number of nodes in the entire hierarchy grows linearly with the number of leaf nodes. Since the leaf nodes recognize events and take actions on the plant (i.e., the outside world), the coordination provided by hierarchical control networks is *scalable*.

There is a mapping from events in the low-level, child controllers to events in the higher-level, parent controller, and a corresponding mapping from controllable events in the parent controller to control patterns in the child controllers. If the mapping from child to parent compresses the data, the

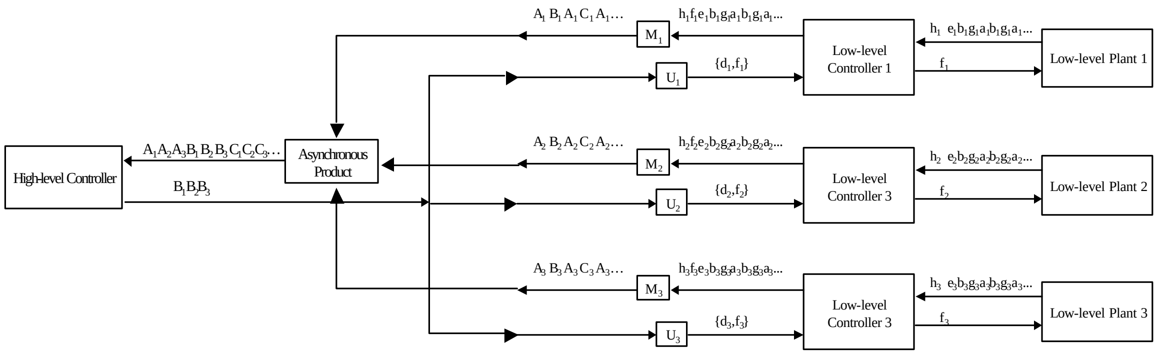


Figure 3. Hierarchical Control of Multiple Low-level Controllers

control structure will exhibit some additional intelligent behaviors. It will appear to *draw conclusions* from lower level events, make decisions based on abstractions, can carry out the decisions at the lower, possible physical, level. There is currently no technique to synthesize or measure "good" aggregations from lower to higher level strings in formal languages. We will therefore use the data compression ratio as an indication of this ability.

Figure 3 illustrates how a high-level controller can control more than one low-level plant. The events being recognized by the high-level controller is the asynchronous product of the high-level symbols being produced in each low-level controller, i.e., they are sent to the high-level controller in the order of their occurrence. Multilevel hierarchies are formed when every non-root node sends higher-level symbols to the level above it, and every non-leaf node sends control patterns to the level below it.

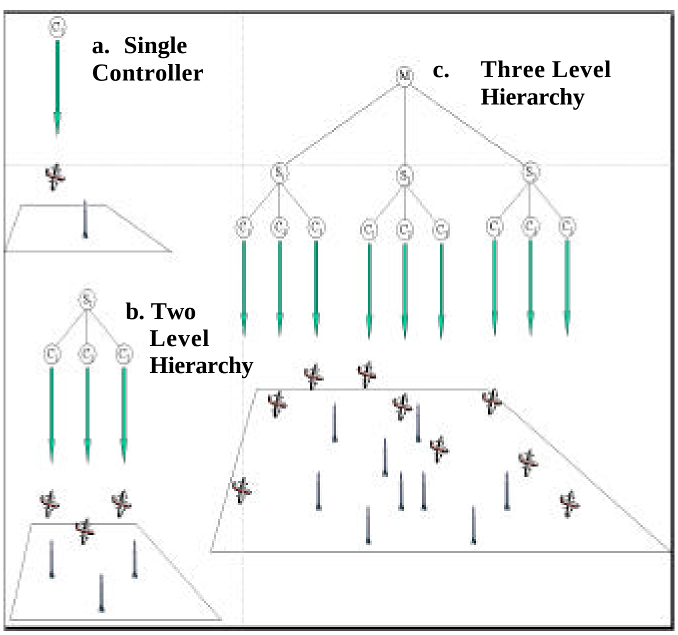


Figure 4. Scalability of DES Controller Hierarchies

The scalability of control hierarchies is shown in Figure 4. The size of the hierarchy increases in proportion to the size of the battlespace. In 4a, a single controller/plane is attacking a single enemy target located in a given area, *A*. In 4b, a two level controller hierarchy with three planes is attacking three enemy targets in an area of *3A*, and in 4c, a three level controller hierarchy with nine planes is attacking nine enemy targets in an area of size *9A*.

5. Conclusions

Intelligent control, using a hierarchy of discrete event controllers, is a good application for deriving quantitative measures of intelligence because these systems are complex enough to exhibit intelligent behavior, but simple enough to allow for a relatively thorough mathematical analysis. Domain independent, quantitative measures of controller performance derived from the analysis are correlated with intelligent behavior by the system. Controllability and hierarchical consistency are correlated with goal-seeking, purposeful behavior; robustness is correlated with the ability to handle uncertainty, and permissiveness is correlated with resourcefulness. The process of aggregation and disaggregation in hierarchical control suggests the ability to make abstractions, plan, and carry out plans.

6. Acknowledgment

Some of the concepts of intelligence metrics presented in this paper were initially developed on the JFACC project, sponsored by the Defense Advanced Research Projects Agency (DARPA) and Air Force Research Laboratory, Air Force Materiel Command, USAF, under agreement number F30602-99-1-0547 (JFACC). The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright annotation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Defense Advanced

Research Projects Agency (DARPA), the Air Force Research Laboratory, or the U.S. Government.

7. References

[1] Barlow, H. B., O*xford Companion to the Mind* (1987).

[2] Baum, A., Newman, S., Weinman, J., West, R. and McManus, *Cambridge Handbook of Psychology Health and Medicine*. Cambridge, Cambridge University Press 1997.

[3] Miele, F., "Skeptic Magazine Interview With Robert Sternberg on The Bell Curve," Skeptic vol. 3, no. 3, 1995, pp. 72-80.

[4] Harnad, S., "Minds, Machines and Searle," *Journal of Theoretical and Experimental Artificial Intelligence* 1: 5-25, 1989.

[5] Delorme M. and Mazoyer, J., *Cellular Automata, A Parallel Model*, Kluwer Academic Publishers, The Netherlands, 1999, pp. 32-35.

[6] Ramadge, P.J. and Wondham, W.M., "Supervisory Control of a Class of Discrete-Event Processes," *SIAM J. Contr. Optim.*, Vol. 25, No. 1, pp. 206-230.

[7] Ramadge P.J. and Wondham, W.M., "The Control of Discrete Event Systems," *Proc. IEEE,* Vol. 77, No. 1, January, 1989.

[8] Ray, A., Xi, W., Zang, H.and Phoha, S., "Hierarchical Consistency of Supervisory Command and Control of Aircraft Operations," *Proc. 2nd Symp. on Advances in Enterprise Control*, Minneapolis, MN, July 10-11, 2000, published by IEEE.

[9] Searle, J. R., "Minds, Brains and Programs," *Behavioral and Brain Sciences 3:* 417-424, 1980.

[11] Takai, S. "*Synthesis of Robust Supervisors for Prefix-Closed Language Specifications*," to be published.

[12] Wondham, W.M. *Notes on Control of Discrete-Event Systems. Systems Control Group, Dept. of Electrical and Computer Engineering*, U. of Toronto, April 1999.

[13] Zhong H. and Wondham, W.M. "On the Consistency of Hierarchical Supervision in Discrete-Event Systems," *IEEE Transactions on Automatic Control*, Vol. 35, No. 10.

Choosing Knowledge Granularity for Efficient Functioning of An Intelligent Agent

Yiming Ye[a] and John K. Tsotsos[b]
IBM T.J. Watson Research Center, Yorktown Heights, NY 10598, USA[a]
Department of Computer Science, York University, Toronto, Ontario, Canada M3J 1P3[b]
(1) 914 784-7460[a] (1) 416 736-2100[b]
yiming@watson.ibm.com[a] tsotsos@vis.toronto.edu[b]

Abstract

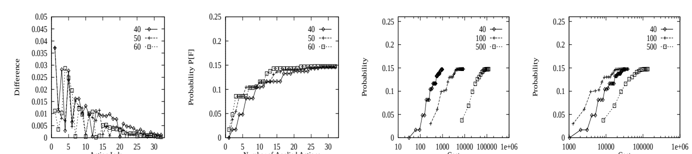
In this paper we introduce the concept of knowledge granularity and study its influence on an agent's action selection process. The goal is to provide a guideline for an agent to select a reasonable knowledge granularity for a given task. Finally we present an idea of using an adaptive mesh method for uneven granularity representation.

1 Introduction

An agent is a computational system that inhabits dynamic, unpredictable environments. It has knowledge about itself and the world. This knowledge can be used to guide its action selection process when exhibiting goal-directed behaviors. Here we address the following question: "How much detail should the agent include in its knowledge representation so that it can efficiently achieve its goal?" There are two extremes regarding granularity of knowledge representation. At one end of the spectrum is the purely reactive scheme which requires little or even no knowledge representation. At the other end of the spectrum is the purely planning scheme which requires the agent to maintain as much detailed knowledge as possible. Experience suggests that neither purely reactive nor purely planning systems are capable of producing the range of behaviors required by intelligent agents in a dynamic, unpredictable environment. This paper offers an alternative point of view of the spectrum of knowledge abstraction based on the granularity of knowledge representation. The goal is to find the proper balance in representing an agent's knowledge such that the representation is detailed enough for the agent to select reasonable actions, and at the same time it is coarse enough that it does not exhaust the agent's resources when selecting those reasonable actions. At the end of the paper, we propose an idea of using adaptive mesh to represent knowledge within a domain.

2 A Case Study

Here we use object search as an example to study the influence of knowledge granularity on the performance of an agent. Object search is the task of searching for a given object in a given environment by a robotic agent equipped with a pan, tilt, and zoom camera [13]. The goal of the agent is to intelligently control the sensing parameters so as to bring the target into the field of view of the sensor and to make the target in the image easily detectable by the given recognition algorithm. To efficiently detect the target, the agent uses its knowledge about the target position to guide its action selection process. This knowledge is encoded as a discrete probability density that is updated whenever a sensing action occurs. To perfectly encode the agent's knowledge, the size of the cube should be infinitely small - resulting in a continuous encoding of the knowledge. But this will not work in general because an infinite amount of memory is needed. In order to make the system work, the agent is forced to represent the knowledge discretely - to use cubes with finite size. This gives rise to an interesting question: how we should determine the granularity of the representation (the size of the cube) such that the best effects or reasonable effects can be generated. The granularity function **G** can be defined as the total memory used by the agent to represent a certain kind of knowledge divided by the memory used by the agent to represent a basic element of the corresponding knowledge. In this case, **G** equals to the total number of cubes in the environment.



(a) (b) (c) (d)

Figure 1: *Experimental results for object search agent.*

We have performed experiments to study the influence of knowledge granularity on the performance of the agent. Usually the higher the value of the knowledge granularity, the longer the time needed to select an action. This is simply because the planning system has more data to be processed. The approximations involved in discretization will cause errors in calculating various values. In general, the higher the value of the knowledge granularity, the less the error caused by discretization. Figure 1(a) shows the errors caused by granularities 40×40, 50×50, and 60×60. The error associated with knowledge granularity may influence the quality of the selected actions, and thus influence the performance of the agent. As shown in Figure 1(b), the higher the granularity, the less the number of actions are needed for the system to reach its detecting limits. Figures 1(c)(d) show the performance of the agent for action execution time 1 second (c) and 1000 second (d), respectively. We can see that a higher granularity may not always beneficial, especially when the action execution time is long.

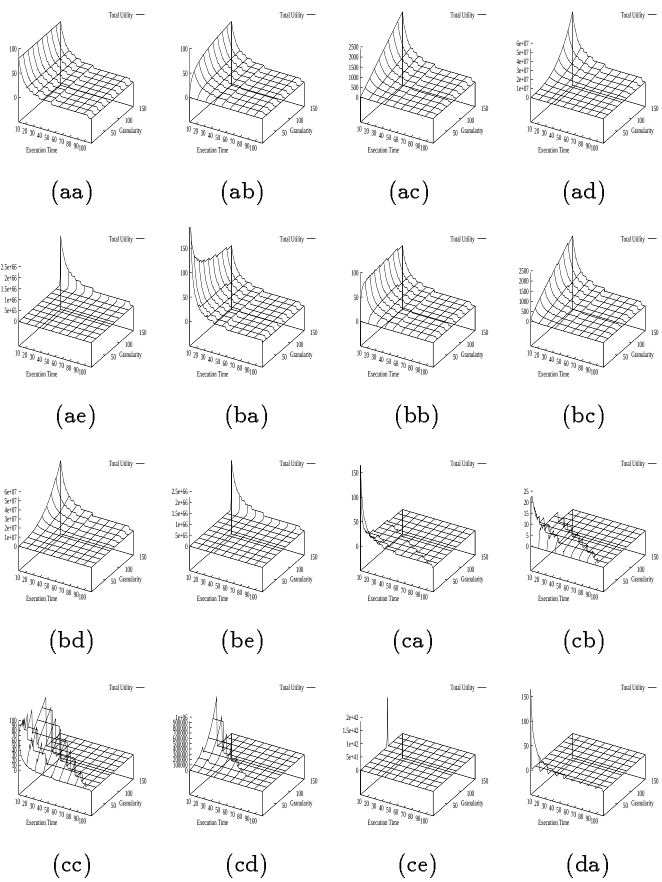


Figure 2: The influence of knowledge granularity on the performance of an agent.

3 Knowledge Granularity in General

In this section, we study in general the influences of knowledge granularity on an agent's action selection performance. For a task oriented agent, a finer granularity usually results in a better selected action. However, the action selection time for a finer granularity is usually longer. Thus, a finer granularity requires more time for selecting actions, and has less time in executing actions. On the other hand, a coarser granularity requires less time for action selection, thus has more time for action execution. In other words, with respect to a fixed time constraint, an agent can usually execute more low quality actions for a coarser granularity, and less high quality actions for a finer granularity. It is thus very interesting to study how the performance of a task oriented agent is influenced by the degree of knowledge granularity, and how the agent should choose a reasonable granularity from the spectrum of knowledge abstraction.

Different agents use different kinds of knowledge and different kinds of action selection procedures. Because of the complexity and diversity of the world of agents, it is impossible to provide a general conclusion or solution with regard to knowledge granularity. What we can do is to group agents into different categories and study the behavior with respect to each category. It is obvious that the performance of an agent is influenced by the action execution time, t_e, the action selection time, t_s, the total time constraint for the given task, T, and the quality Q of the selected and executed actions. t_s and Q is influenced by the knowledge granularity adopted by the agent. Suppose for a granularity g, the *average* time needed in selecting an action is $t_s(g)$, the *average* contributions of a selected action to the task is $Q(g)$. *Assuming* that the total contributions U made by an agent within the time constraint T can be represented by the sum of the average contributions of all the actions that is executed within T. Then, U can be represented as follows.

$$U(g) = \lfloor \frac{T}{t_e + t_s(g)} \rfloor Q(g)$$

In the following, we study how $U(g)$ is influenced by different $t_s(g)$ and $Q(g)$. We assume $T = 100$ and $g \in [1, 150]$.; The following functions are used in our empirical study: $f_a(g) = 5$; $f_b(g) = ln(g)$; $f_c(g) = g + 1$; $f_d(g) = g * g * g + 1$; $f_e(g) = exp(g) + 1$. These functions represent different relations between the knowledge granularity g and the entities to be discussed. Function $f_a(g)$ means that the entity is a constant, and thus is not influenced by granularity. $f_b(g)$, $f_c(g)$, $f_d(g)$, $f_e(g)$ refer to different degrees of the influence of granularity on the entity. Figure 2 shows how the granularity influences the performance of the agent under different situations. The graphs are indexed by the above functions. For example, Figure 2(ab) corresponds to the situations that $t_s(g) = f_a(g)$ and $Q(g) = f_b(g)$.

From Figure 2, we can notice that t_e is a very important factor that influences the selection of the knowledge granularity. Figures 2(aa)(ab)(ac)(ad)(ae) show the situation when t_s is not influenced by the granularity. In this special case, the finer the granularity, the better the performance, except for the first one (aa). From Figures 2 (ca)(cb)(cc)(cd)(ce) we can notice that for a large execution time (≥ 50), g should be low in order to guarantee that at least one action can be executed. Figure 2(ca) shows the situation that the benefit of each action is not influenced by the granularity, thus a smaller granularity is preferred no matter what the action execution time is. From Figure 2(cc) we can notice that the situation becomes complex. For example, for a small t_e, there are several granularities that can generate satisfactory results. These reasonable granularities are different for different t_e.

4 Selecting Knowledge Granularity

The experiments in the above section show that the level of knowledge granularity has a big impact on the quality and speed of the agent's behavior. It is thus important for an agent to adapt its knowledge granularity based on environmental and task-specific demands. In this section, we address the following interesting question: how can we select the knowledge granularity $\mathbf{G(k)}$ for a given representation scheme k such that the best agent performance or a relatively good agent performance can be achieved?

In some situations, we are able to select the best knowledge granularity in the sense that it maximizes the performance of the agent. Here is an example. Suppose we have an agent whose task is to collect food from a region of length L within a time limit T. The agent can use different representation lengths $\Lambda = \{l_1, \ldots, l_q\}$ to represent the region. (suppose $\frac{L}{l_i}$ is integer, where $1 \leq i \leq q$). If the agent selects $l \in \{l_1, \ldots, l_q\}$ as its representation scheme $\mathbf{k}$ for the corresponding knowledge, then the corresponding knowledge granularity for this scheme will be $\mathbf{G}(\mathbf{k}) = \frac{L}{l}$. The total region is thus divided into $\frac{L}{l}$ units. The process of food collection is as follows. Before the collecting process, all the units of the region will be in the status of "not ready". When the collecting process begin, one of the units becomes "ready". The agent will then search for this unit. The time, $t_s(l)$, used by the agent to locate the unit is the time for the agent to select an action under the current representation scheme. Suppose $t_s(l) = \frac{1}{l}$. After the unit is located, the agent will collect food from this unit. The total time needed for the agent to collect food is the time needed for the agent to execute the selected action. Suppose it is $t_e(l) = Cl$ (where C is a constant). The total amount of food that is collected is $B(l) = \frac{1}{l}$. When the agent finishes its food collection process at the selected unit, the status of another unit will become "ready". The agent will search for this new unit and collect food again from this new unit. This process will continue until the total time T is used up. If the total time T is exhausted when the agent is locating a unit or when the agent is collecting food within a unit, then the amount of collected food from the corresponding unit will be zero. It is obvious that the number of units that can be processed by the agent within T is $\frac{T}{t_s(l)+t_e(l)}$, and the number of units available is $\frac{L}{l}$.

The performance $\mathbf{P}$ of the agent is measured by the total amount of food collected by the agent and is given by the following formula:

$$\mathbf{P} = \begin{cases} \frac{L}{l}B(l) & \text{if } \frac{T}{t_s(l)+t_e(l)} \geq \frac{L}{l} \\ \lfloor \frac{T}{t_s(l)+t_e(l)} \rfloor B(l) & \text{if } \frac{T}{t_s(l)+t_e(l)} < \frac{L}{l} \end{cases} . \quad (1)$$

This is actually

$$\mathbf{P} = \begin{cases} \frac{L}{l^2} & \text{if } l \leq \sqrt{\frac{L}{T-CL}} \\ \lfloor \frac{Tl}{Cl^2+1} \rfloor \frac{1}{l} & \text{if } l > \sqrt{\frac{L}{T-CL}} \end{cases} . \quad (2)$$

The problem is to find a l in $\Lambda = \{l_1, \ldots, l_q\}$ such that $\mathbf{P}$ is maximized. The set Λ can be divided into two parts $\Lambda_A = \{l_1, \ldots, l_j\}$ and $\Lambda_B = \{l_{j+1}, \ldots, l_q\}$, such that all the elements in Λ_A are less than $\sqrt{\frac{L}{T-CL}}$, and all the elements in Λ_B are greater than or equal to $\sqrt{\frac{L}{T-CL}}$. It is obvious that for elements $l \in \Lambda_A$, the smallest one has the best performance because $\frac{L}{l^2}$ is a decreasing function. For elements $l \in \Lambda_B$, we can calculate the value of $\lfloor \frac{Tl}{Cl^2+1} \rfloor \frac{1}{l}$ to identify the best element. Then we compare the smallest element in Λ_A and the best element in Λ_B to identify the one that maximizes the performance of the system.

The above example shows that in some situations, an agent is able to identify an optimum knowledge granularity based on the task requirement (here T) and the environmental characteristics (here L). The basic method is to try to represent the performance of the agent as a function of the agent's knowledge granularity, and then to find the granularity that maximizes the performance.

In general, it is very difficult or even impossible to find a best knowledge granularity for an agent, because the performance of the agent might be influenced by many other factors in addition to the knowledge granularity. For example, there does not exist a best knowledge granularity for the object search agent, because its performance is also influenced by the initial target distribution. A granularity that is best for one distribution might not be the best for another distribution. Thus, in general, we need to relax our requirements. Instead of finding the best granularity, we search for a reasonable one such that a relatively good performance can be achieved. Because of the variations of different agent systems, it is impossible to provide a detailed procedure to select the acceptable granularity that can be applied to all the agent systems. However, we can provide a general guideline for the selection of the knowledge granularity.

5 Selecting Reasonable Granularity in Complex Agent Environment

In an agent environment where the relationships among the task constraints, the environments, and the knowledge granularity are very complex, the "demand-environment-granularity" (DEG) Hash Table can be used to select a reasonable granularity. The DEG Hash Table is a Hash Table such that the "key" is the combination of different factors and the "value" is the granularity that is appropriate for the corresponding factors. When an agent is informed of task requirements, it first transforms the task requirements and the environmental factors into a key. Then it retrieves the granularity from the DEG Hash Table based on the key. This granularity will be used by the agent to represent the corresponding knowledge.

For a complex agent environment, it might have more than one task constraints $T_1, \ldots, T_{n_T}$. Each $\mathbf{T}_\mathrm{i}$ forms one component in the "key" of the DEG Hash Table. It can be divided into several groups $T_{i,1}, \ldots, T_{i,k}$ based on certain criteria. For example, the task constraint for an object search agent is the total time available for the search. This time constraint can be divided into groups like "from 1 second to 30 seconds", "from 30 seconds to 100 seconds", etc..

In addition to the task constraints, we should also consider the influences of the environmental factors when selecting the granularity. Suppose $E_1, \ldots, E_{n_E}$ are the environment factors that need to be considered. Like above, each $\mathbf{E}_\mathrm{i}$ can be divided into several groups $T_{i,1}, \ldots, T_{i,k}$ based on a certain criteria.

The DEG Hash Table is then looks like following:

$\mathbf{T}_1$	...	$\mathbf{T}_{\mathbf{n_T}}$	$\mathbf{E}_1$	...	$\mathbf{E}_{\mathbf{n_E}}$	$\mathbf{G}$
t_1	...	t_{n_T}	e_1	...	e_{n_E}	g
⋮	⋮	⋮	⋮	⋮	⋮	⋮

Table 1

Where each row in the table, except the first one, gives a "key" $(t_1, \ldots, t_{n_T}, e_1, \ldots, e_{n_E})$ and the corresponding granularity value g. Here, t_i is a category (group) for the task constraint factor $\mathbf{T}_\mathrm{i}$ and e_i is a category (group) for the environmental factor $\mathbf{E}_\mathrm{i}$. Term g is the knowledge granularity value corresponding to the "key" and should be obtained by conducting various simulation experiments or theoretical analysis before the agent performs any task. When an agent is informed of a task, it first determines the key based on the current situations, and then uses this "key" to locate the knowledge granularity.

6 The Adaptive Mesh Approach

The above analysis and methods assumes that an agent maintains exactly the same granularity for the whole region. This may not be necessary. For example, for the case of object search, the search agent may only need to have a detailed representation of the surrounding area or areas with high density. Because these areas are the most important areas that need to be considered during the sensor planning process. Thus, the granularity should be different for different areas. Actually people seldom maintain the same granularity when they perform tasks. They dynamically adjust the granularity at different time and under different situations. For example, when a person travels from his university to another city to attend a conference, the representation of the geographical situations or maps will be different at different times and context. Before he left for the conference, he will have more detailed representation of his office and less detailed representation of the conference site. However, when the plane is about to arrive at the destination, his representation of the conference site will be much more detail and he will intentionally use more crude representation for his office.

The above discussion suggests that it might be beneficial for an agent to maintain a non-uniform representation of its knowledge and dynamically adjust the granularity distribution based on context. In the following, we propose a method of achieving this. We illustrate our approach under the scenario of object search. Our goal is to adjust the representation density based on the target distribution, the higher the density, the more detail the representation.

Our approach is to first obtain the highest granularity map as represented by little cubes. Then, we combine those cubes whose probabilities are not big enough into a larger blob. This process will continue until the probability within the blob is big enough. This effort will result in an mesh representation of the knowledge.

Here is the algorithm:

1. Tessellate the region into small cubes corresponding to the highest granularity.
2. Assign all the small cubes as unconsidered.
3. The process terminates when all the cubes are considered.
4. Find the unconsidered cube with the smallest probability, and mark it as considered.
5. If the probability is smaller than the threshold, then find a neighbor cube that is unconsidered with the smallest probability. Mark this cube as considered. If there is no such cubes, goto 3.
6. Combine the two together and form the blob. If the probability of the blob is bigger than the threshold, goto 3.
7. Find a neighbor unconsidered cube of the blob with the smallest probability. Mark it as considered and goto 6.

7 Conclusion

The message derived from both the case study and the general analysis is that knowledge granularity has a big impact on the performance of an agent. Thus, an appropriate knowledge granularity should be selected by an agent in order to guarantee a satisfactory result. In complex situations, the selection of granularity depends on many factors. In general, we can construct graphs like Figure 2 to analyze the effects of granularity on the performance of the agent under different factors and constraints, and then select a favorable granularity. We may also use an adaptive mesh to represent knowledge in some situations.

References

[1] R. Bajcsy. Active perception vs. passive perception. In *Third IEEE Workshop on Vision*, pages 55–59, Bellaire, 1985.

[2] R. A. Brooks. Intelligence without representation. *Artificial Intelligence*, (47):139–160, 1991.

[3] I. Ferguson. *Touring Machine: An Architecture for Dynamic, Rational, Mobile Agents.* PhD thesis, University of Cambridge, UK, 1992.

[4] T. D. Garvey. Perceptual strategies for purposive vision. Technical Report Technical Note 117, SRI International, 1976.

[5] M. Georgeff and A. Lansky. Reactive reasoning and planning. In *Proceedings of the Sixth National Conference on Artificial Intelligence*, pages 677–682, Seattle, WA, 1987.

[6] A. Meystel. Theoritical foundations of planning and navigation for autonomous mobile systems. *International Journal of Intelligent Systems*, 2(2):73–128, 1987.

[7] A. Meystel. Multiresolutional architectures for autonomous systems with incomplete and inadequate knowledge representation. In *Artificial Intelligence in Industrial Decision Making, Control, and Automation. Eds. S.G. Tzafestas and H.B. Verbruggen. Kluwer Academic*, pages 159–223, 1995.

[8] A. Meystel, R. Bhatt, D. Gaw, P. Graglia, and S. Waldon. Multiresolutional pyramidal knowledge representation and algorithmic basis of imas-2. In *Proc. of Mobile Robots, SPIE Vol. 851*, pages 80–116, 1987.

[9] J. Muller and M. Pischel. Modelling interacting agents in dynamic environments. In *Proceedings of the Eleventh European Conference on Artificial Intelligence*, pages 709–713, Amsterdam, The Netherland, 1994.

[10] S. Sen, S. Roychowdhury, and N. Arora. Effects of local information on group behavior. In *Proceedings of Second International Conference on Multi-Agent Systems*, pages 315–321, Kyoto, Japan, 1996.

[11] T. Tyrrell. *Computational Mechanisms for Action Selection.* PhD thesis, Center for Cognitive Science, University of Edinburgh, England, 1993.

[12] M. Wooldridge and N. Jenning. Intelligent agents: theory and practice. *The Knowledge Engineering Review*, 10(2):115–152, 1995.

[13] Y. Ye. *Sensor Planning for Object Search*. PhD thesis, University of Toronto, Toronto, Canada, January 1997.

[14] Y. Ye and J. K. Tsotsos. Sensor planning in 3d object search: its formulation and complexity. In *The 4th International Symposium on Artificial Intelligence and Mathematics*, Florida, U.S.A., January 3-5 1996.

[15] Y. Ye and J. K. Tsotsos. Knowledge difference and its influence on a search agent. In *First International Conference on AUTONOMOUS AGENTS*, Marina del Rey, CA, January 1997b.

[16] Y. Ye and J. K. Tsotsos. Knowledge granularity and action selection. In *Eighth International Conference on Artificial Intelligence: Methodology, Systems, and Applications*, pages 475–489, 1998.

[17] Y. Ye and J. K. Tsotsos. Sensor planning for 3d object search. *Computer Vision and Image Understanding*, 73(2):145–169, February 1999.